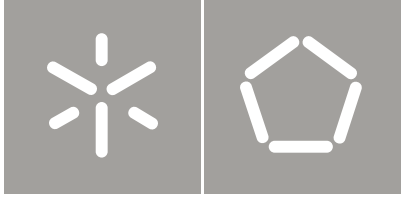
Universidade do Minho
Escola de Engenharia

Álvaro Miguel de Oliveira Araújo

Sistema Robotizado para Execução Automática do Processo de Halogenação para a Indústria do Calçado

Outubro de 2010

Sistema Robotizado para Execução Automática do Processo de Halogenação para a Indústria do Calçado
Álvaro Miguel de Oliveira Araújo
UMinho | 2010



Universidade do Minho
Escola de Engenharia

Álvaro Miguel de Oliveira Araújo

Sistema Robotizado para Execução Automática do Processo de Halogenação para a Indústria do Calçado

Tese de Mestrado
Ciclo de Estudos Integrados Conducentes ao
Grau de Mestre em Engenharia Electrónica Industrial e Computadores

Trabalho efectuado sob a orientação do
Professor Doutor Jaime Francisco Cruz Fonseca

Outubro de 2010

Agradecimentos

Aos meus pais Manuel e Manuela, à minha irmã, Inês e madrinha, Maria pelo apoio e incentivo.

À minha namorada, Sofia por toda a compreensão e paciência.

Aos professores e técnicos do curso de Electrónica Industrial e Computadores pelo conhecimento e estímulo proporcionados, em especial,

Ao meu orientador, Prof. Dr. Jaime Francisco Cruz Fonseca, pela disponibilidade e oportunidade de poder realizar esta dissertação de mestrado.

A todos os meus colegas de curso, pelo acompanhamento académico e pessoal que me foi concedido ao longo dos anos.

Aos familiares e amigos não mencionados.

Resumo

Devido à enorme competitividade existente no mercado industrial torna-se inevitável a extrema exigência no que diz respeito à qualidade, tempo de resposta e, principalmente, à inovação de novas técnicas de produção para uma melhor satisfação do consumidor.

Actualmente a indústria do calçado depara-se com múltiplos desafios. Em primeiro lugar, os consumidores exigem cada vez mais novos produtos mais confortáveis e novo *design*; em segundo lugar a concorrência no mercado global é cada vez mais forte. A indústria do calçado tradicional apresenta bons níveis de desempenho contudo necessita de aumentar os níveis de automação para que se torne ainda mais competitiva.

O trabalho proposto para esta dissertação tem como objectivo optimizar e automatizar um dos processos mais utilizados neste tipo de indústria: limpeza das solas (halogenação das solas). Este processo está sempre presente no fabrico do calçado sendo feito de uma forma manual ou por um sistema semi-automatizado muito simples. O processo de halogenação é efectuado antes da aplicação da cola. O material químico usado é tóxico e prejudicial para a saúde.

Palavras-chave:
Robótica, processamento de imagem

Abstract

Due to the enormous competition in the industrial market it becomes inevitable the extreme demanding of quality, time response and mainly developing new production techniques for a better consumers satisfaction.

Nowadays the shoemaking industry faces multiple challenges. In the first place consumers require more comfortable and better designed products; in second place competition is getting stronger as each day goes by. The traditional shoemaking industry presents good levels of performance; nevertheless it needs to increase its automation levels in order to become more competitive.

The main objective of this dissertation is to optimize one of the most used processes in shoemaking industry: the cleaning of the shoe soles (halogenation). This process is always present in the shoemaking industry but it is performed manually or by a very simple semi-automatic system. The halogenation process is made before the glue application in the soles. The material used in this process is toxic and highly harmful to health.

Keywords:
Robotics, image processing

Índice

Capítulo 1 1

1 Introdução 1

1.1 Motivação do trabalho 3

1.2 Objectivos do trabalho 4

1.3 Organização da dissertação 4

Capítulo 2 7

2 Estado da arte 7

2.1 Processos de fabrico de calçado 7

2.1.1 *Welt Sewing Machine* 8

2.1.2 *Thermoadesive Attaching Machine* 8

2.1.3 *Inseam Trimming Machine* 9

2.2 Robótica no fabrico de calçado 10

2.3 Visão por computador no fabrico de calçado 12

2.4 Conclusões 13

Capítulo 3 15

3 Sistema de visão por computador 15

3.1 Iluminação 15

3.2 Câmara CCD Firewire DFK21F04 17

3.3 Técnicas de processamento de imagem 19

3.3.1 Calibração 19

3.3.2 Conversão binária ou *Thresholding* 22

3.3.3 Filtragem 24

3.3.4 Erosão 26

3.3.5 Detecção de contornos 27

3.3.6 Obtenção dos pontos dominantes 31

3.3.7 Transformação dos pontos em coordenadas reais 35
3.4 Conclusões 37
Capítulo 4 39
4 Aplicação em Labview 39
4.1 Código do sistema de visão por computador 40
4.2 Protocolo de comunicação 44
4.2.1 Configuração da porta série 44
4.2.2 Protocolo de envio 45
4.3 Conclusões 48
Capítulo 5 50
5 Sistema robótico 50
5.1 Aspectos teóricos 50
5.2 Modos de programação 51
5.2.1 Programação *On-line* 51
5.2.2 Programação *Off-line* 53
5.3 Software do manipulador 56
5.3.1 Algoritmo 56
5.3.2 Código do manipulador 58
5.4 Conclusões 60
Capítulo 6 63
6 Resultados e Conclusões 63
6.1 Resultados 63
6.1.1 Interface LabView 63
6.1.2 Sistema robótico 67
6.2 Conclusões 69
6.2.1 Trabalho futuro 70

Lista de figuras

Figura 1.1 - Zona de aplicação do material químico 2
Figura 1.2 - Diferentes tipos de solas 2
Figura 1.3 - Esquema ilustrativo da implementação do sistema 3
Figura 2.1 - Máquina de utilizada na costura de calçado (Welt Sewing Machine) 8
Figura 2.2 - Máquina de colocar adesivo nas solas (Thermoadesive Attaching Machine) 9
Figura 2.3 - Máquina de corte (Inseam Trimming Machine) 10
Figura 2.4 - Braço robótico utilizado no transporte de calçado para a máquina [7] 11
Figura 2.5 - Braço robótico a executar tarefa numa sola de sapato [7] 11
Figura 2.6 - Simulador utilizado pela Lirel [7] 12
Figura 3.1 - Técnicas de iluminação; (a) - Iluminação frontal; (b) - Retro Iluminação . 16
Figura 3.2 - Imagem do calcanhar da sola sem iluminação 17
Figura 3.3 - Imagem do calcanhar da sola com iluminação 17
Figura 3.4 - Exemplo de um CCD [11] 18
Figura 3.5 - Câmara DFK21F04 18
Figura 3.6 - Diagrama funcional das técnicas de visão implementadas 19
Figura 3.7 - Distorção para exterior (à esquerda) e para o interior (à direita) [13] 20
Figura 3.8 - Calibração da câmara 20
Figura 3.9 - Dimensões da imagem 22
Figura 3.10 - Intensidade dos pixéis em tons de cinzento (0 - preto até 255 - branco) [14] 23
Figura 3.11 - Imagem da sola e respectivo histograma 23
Figura 3.12 - Imagem binária 24
Figura 3.13 - Exemplo de filtragem de uma imagem [15] 25
Figura 3.14 - Imagem original da sola à esquerda e imagem filtrada à direita 26
Figura 3.15 - Exemplo de erosão de um objecto circular e respectivas matrizes: (a) imagem original; (b) imagem erodida na orientação pretendida; (c) imagem erodida na orientação pretendida. 26
Figura 3.16 - Matriz de erosão 27
Figura 3.17 - Imagem original à esquerda e imagem erodida à direita 27
Figura 3.18 - Exemplo de aplicação de filtros de detecção de contorno; (a) - Roberts; (b) - Sobel; (c) - Prewitt 30

Figura 3.19 - Contorno da sola 30
Figura 3.20 - Exemplo de extracção de pontos dominantes de diferentes objectos; (a) - Quadrado; (b) - Circulo 31
Figura 3.21 - Ilustração do algoritmo das tangentes; (a) - Espaçamento maior entre os pontos; (b) Espaçamento menor entre os pontos. 32
Figura 3.22 - Exemplo de algoritmo de aproximação poligonal; (a) - segmento de recta entre ponto inicial e final; (b) - marcação do primeiro ponto; (c) - marcação do segundo ponto [1] 33
Figura 3.23 - (a) - Número de pixéis do contorno; (b) - Número de pixel actual; (c) - Curvatura do pixel actual em relação ao anterior. 34
Figura 3.24 - Resultado dos pontos dominantes com valor máximo de 0,4 35
Figura 3.25 - Resultado dos pontos dominantes com valor máximo de 0,2 35
Figura 3.26 - *Array* de coordenadas pixel dos pontos dominantes 36
Figura 3.27 - Pontos dominantes reais 37
Figura 4.1 - Exemplo de interface LabView; (a) - Painel frontal; (b) - Diagrama de blocos 40
Figura 4.2 - Referência da câmara 40
Figura 4.3 - Diagrama de blocos para adquirir a imagem 41
Figura 4.4 - Código das técnicas de processamento de imagem 41
Figura 4.5 - Interface de calibração da imagem 42
Figura 4.6 - Introdução da dimensão da matriz 42
Figura 4.7 - Tipo de operação e número de iterações 42
Figura 4.8 - Filtro de detecção de contornos 42
Figura 4.9 - Imagem de código da obtenção de pontos dominantes; (a) - Obtenção de todos os pontos de contorno em array; (b) - Disponibilização do valor da curvatura; (c) - Soma da curvatura com valor anterior; (d) - Teste ao valor acumulado; (e) – Marcação do ponto 43
Figura 4.10 - Código da transformação dos pontos para o mundo real 44
Figura 4.11 - Configuração da porta série 44
Figura 4.12 - Código da configuração da porta série; (a) - Bloco de abertura da porta série 45
Figura 4.13 - Fluxograma do protocolo de comunicação 46
Figura 4.14 - Código do protocolo de envio dos pontos dominantes; (a) – Bloco de leitura; (b) – Bloco de escrita; (c) - Bloco de leitura 47

Figura 4.15 - (a) - Bloco de emparelhamento que permite formatar as duas coordenadas em string; (b) - Coordenadas enviadas com caracteres de controlo 47
Figura 5.1 - Imagem do interface Homem/Robot 51
Figura 5.2 - Software RobotStudio On-line 52
Figura 5.3 - Lista de instruções 52
Figura 5.4 - Software RobotStudio 5.12.03 53
Figura 5.5 - Código introduzido no robot virtual 54
Figura 5.6 - Resultado do código introduzido no robot 55
Figura 5.7 - Flexpendant virtual 55
Figura 5.8 - Trama a enviar para cada ponto 56
Figura 5.9 - Funcionamento do software do robot 57
Figura 6.1 - Aquisição da imagem 64
Figura 6.2 - Conversão binária 64
Figura 6.3 -Detecção de contornos 65
Figura 6.4 - Erosão 65
Figura 6.5 - Marcação de pontos dominantes 66
Figura 6.6 - Ambiente utilizado 67
Figura 6.7 - Robot na posição inicial 68
Figura 6.8 - Execução da tarefa 68
Figura 6.9 - Execução da tarefa 69
Figura 7.1 - Controlador IRC 5 e Flexpendant 76
Figura 7.2 - Mesa de trabalho 76
Figura 7.3 - Robot IRB140 77
Figura 7.4 - Suporte de iluminação, câmara e sola de sapato 77
Figura 7.5 - Função "Open" [19] 78
Figura 7.6 - Função "write" [19] 78
Figura 7.7 - Função "readstr" [19] 79
Figura 7.8 - Função "strmatch" [19] 79
Figura 7.9 - Função "strpart" [19] 80
Figura 7.10 - Função "strtoval" [19] 80
Figura 7.11 - Função "movel" [19] 81
Figura 7.12 - Função "offs" [19] 81
Figura 7.13 - Especificações do robot IRB 140 [20] 82
Figura 7.14 - Alcance máximo de cada eixo do robot [20] 82

Figura 7.15 - Especificações do controlador IRC5 [21] 83
Figura 7.16 - Dimensões do controlador [21] 83
Figura 7.17 - Robot IRB 140 da ABB 84
Figura 7.18 - Eixos do robot 85
Figura 7.19 - Controlador IRC 5 87
Figura 7.20 - Botões de módulo de controlo 87
Figura 7.21 - Chave de modo 88
Figura 7.22 - Imagem do interior do armário 89
Figura 7.23 - Flexpendant 90
Figura 7.24 - Acessórios da Flexpendant [22] 90
Figura 7.25 - Botões da Flexpendant [22] 91
Figura 7.26 - Ecrã inicial Flexpendant [22] 92
Figura 7.27 - Opções do menu ABB 93
Figura 7.28 - Editor de programa 93

Lista de tabelas

Tabela 3.1 - Performance 18
Tabela 3.2 - Interface eléctrico 19
Tabela 3.3 - Dimensões e peso 19
Tabela 7.1 - Ângulos máximos de cada eixo [20] 85
Tabela 7.2 - Velocidades dos eixos do robot [20] 86
Tabela 7.3 - Dimensões do robot [20] 86
Tabela 7.4 - Botões do módulo de controlo [22] 88
Tabela 7.5 - Legenda da chave de modo 88
Tabela 7.6 - Legenda dos acessórios da Flexpendant [22] 90
Tabela 7.7 - Legenda dos botões da Flexpendant [22] 91
Tabela 7.8 – Legenda do ecrã inicial da Flexpendant 92

Abreviaturas e Símbolos

CCD - *Charge-couple device*

ABB - *Asea Brown Boveri*

RS 232 - *Recommended standard* 232

rpm - Rotações por minuto

NC - *Numerical control*

CNC - *Computer numerical control*

CAD - *Computer-aided design*

CAM - *Computer-aided manufacturing*

PUMA – *Programmable universal manipulation arm*

PC – *Personal computer*

FTP – *File Transfer Protocol*

USB – *Universal Serial Bus*

∇ - Gradiente

Capítulo 1

1 Introdução

Hoje em dia existe um grande número de empresas que utilizam como meio de produção alguns processos rudimentares, empregando pessoas em ambientes fabris perigosos e prejudiciais para a sua saúde.

Contrariamente, a indústria do calçado é das mais fortes e das que mais aposta na inovação dos seus processos de fabrico, pois os seus consumidores necessitam de produtos cada vez mais confortáveis e de novo *design* [1]. A concorrência neste mercado é elevada, como se pode verificar pelo grande número de modelos e marcas de calçado existentes. Logo, uma das formas de rentabilizar e optimizar esta indústria seria automatizar ao máximo os seus processos de produção em proveito de uma maior competitividade e produtividade, e consequentemente um melhor desenvolvimento económico e social.

Na produção de calçado é, inevitavelmente, utilizado um processo que é denominado de halogenação. Este processo consiste na limpeza das solas, mais concretamente da região de colagem desta à pele de revestimento do sapato. Essa região localiza-se normalmente a uma distância de aproximadamente 0,5 cm dos limites exteriores da sola (Figura 1.1). A limpeza é efectuada através da aplicação de um produto químico por fricção na zona anteriormente referida para que a sola fique devidamente limpa de impurezas, aumentando desta forma, a eficiência da colagem desta à pele do futuro sapato.

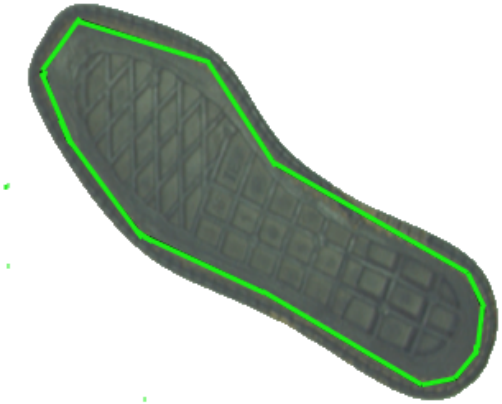


Figura 1.1 - Zona de aplicação do material químico

Com a utilização de produtos químicos, torna-se indispensável que os trabalhadores utilizem equipamento de protecção apropriado, como por exemplo luvas e máscaras e que tenham muito cuidado aquando da sua manipulação e aplicação, pois existem riscos para a saúde [2].

Para ultrapassar estas dificuldades e aumentar a eficiência e a uniformidade do processo de limpeza da sola foi desenvolvido um sistema completamente automático para realizar esta tarefa. O sistema baseia-se na utilização de técnicas de processamento de imagem 2D para identificação dos diferentes formatos de solas (Figura 1.2) e extracção dos pontos relevantes necessários para descrever a região a ser limpa. Para efectuar a tarefa de limpeza foi utilizado um braço robótico antropomórfico.

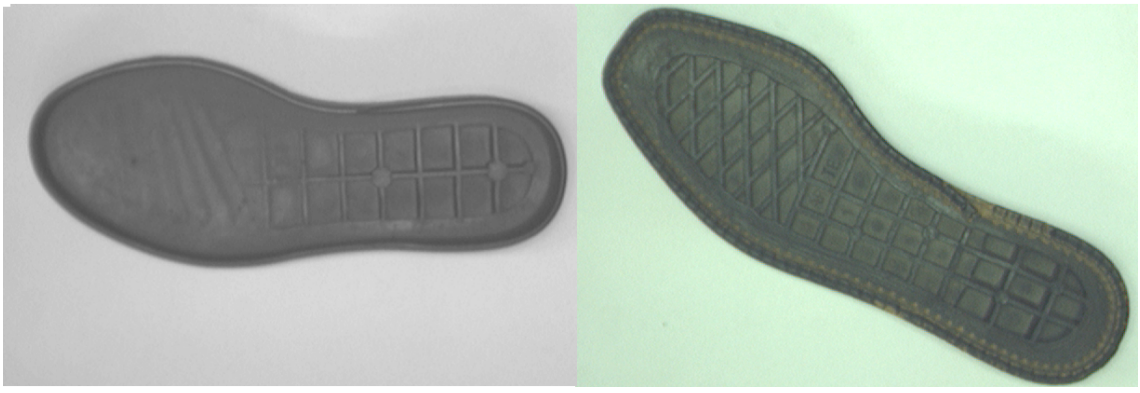


Figura 1.2 - Diferentes tipos de solas

A figura 1.3 mostra um esquema geral do sistema em que é utilizada uma câmara CCD que fará a aquisição da imagem para posterior processamento. A finalidade do processamento de imagem será extrair os pontos relevantes, que serão enviados para o robot para este os percorrer, efectuando a tarefa de limpeza da sola.

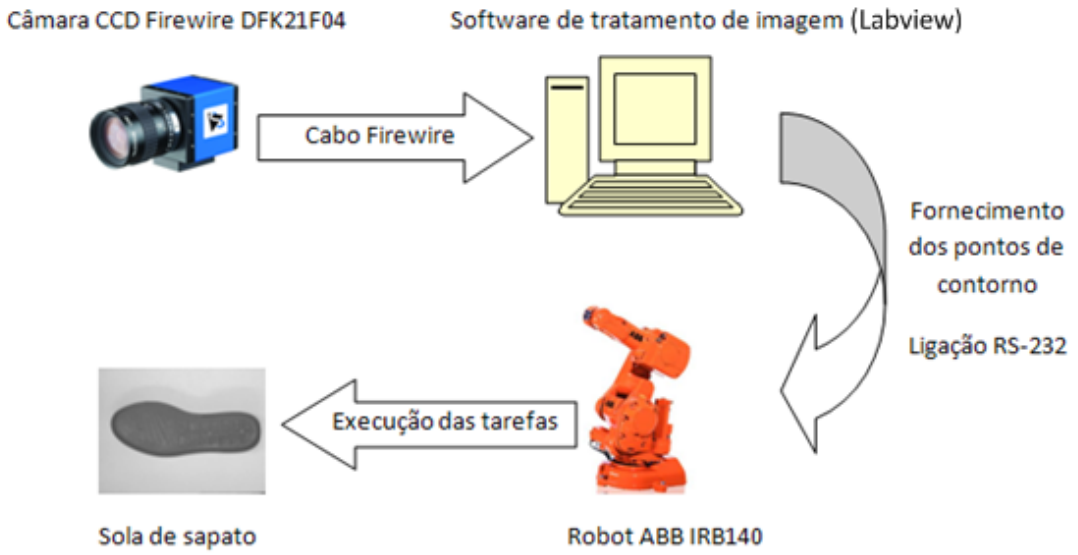


Figura 1.3 - Esquema ilustrativo da implementação do sistema

1.1 Motivação do trabalho

A principal motivação foi a procura de uma solução protótipo para um problema já antigo na indústria do calçado que é a halogenação das solas dos sapatos e que por diferentes motivos ainda não se encontrou uma solução definitiva para o mesmo, apesar dos riscos que comporta para a saúde do operário que executa a tarefa.

Por outro lado a possibilidade de trabalhar com um robot industrial equipado com um controlador de última geração e a integração deste com um sistema de visão por computador foi uma motivação extra.

1.2 Objectivos do trabalho

O principal objectivo deste trabalho de mestrado foi o desenvolvimento de um sistema robotizado para efectuar o processo de halogenação nas solas.

Para se atingir os objectivos pretendidos o trabalho foi dividido nas seguintes fases:

- Desenvolvimento de uma aplicação em Labview que permita identificar as solas e extrair os pontos relevantes para o processo de halogenação das solas. Para isso, foram estudadas e aplicadas técnicas de processamento de imagem como por exemplo:
    - Calibração
    - Conversão binária ou *Thresholding*
    - Filtragem
    - Erosão
    - Detecção de contornos
    - Extracção dos pontos relevantes das solas para posterior envio para o robô.

- Desenvolvimento de uma aplicação de software que permita a transformação dos pontos de contorno extraídos da imagem em coordenadas reais, visto que os pontos retirados são em coordenadas pixel.
- Estudo do manipulador robótico ABB.
- Programação automática do manipulador industrial ABB para execução do processo de halogenação.

1.3 Organização da dissertação

Os temas abordados nesta dissertação estão organizados por capítulos e da seguinte forma:

No segundo capítulo é descrito o estado da arte.

No terceiro capítulo são apresentados os conceitos teóricos da visão por computador.

No quarto capítulo é descrita a aplicação em LabView. Aqui é realizada uma explicação detalhada do código das técnicas de processamento de imagem, porta série e transmissão dos pontos do contorno da sola para o robot.

O quinto capítulo é introdutório ao robot antropomórfico, aqui serão explicados os conceitos de programação do robot, e todo o software envolvido na recepção dos pontos da sola e execução da tarefa de halogenação.

No sexto e último capítulo são apresentados os resultados dos interfaces realizados em LabView, processamento de imagem e realização da tarefa de halogenação. Neste capítulo são, também, discutidas as conclusões finais e perspectivas futuras da dissertação.

Capítulo 2

2 Estado da arte

Ao longo dos tempos o calçado fez parte do dia-a-dia do homem. Primeiramente, foi utilizado como protector e aquecedor mas actualmente usa-se, essencialmente, como adorno e acessório de moda.

Cada vez mais a máquina substitui o homem na indústria e este mercado não é excepção, apesar de trazer consequências sociais, a máquina consegue fazer um trabalho mais rápido, eficiente e barato que o homem. Assim, a automatização de processos no fabrico de calçado significa a evolução de qualquer entidade que os aplique.

Ao longo deste capítulo serão apresentados alguns processos de fabrico de calçado, assim como a utilização da visão por computador e da robótica na produção de calçado.

2.1 Processos de fabrico de calçado

Actualmente o fabrico de calçado está muito desenvolvido comparativamente com o século XIX e inícios do século XX, devido à introdução de novas máquinas revolucionou-se a produção e a qualidade do calçado produzido.

A produção de calçado é realizada através da execução de algumas etapas de preparação dos materiais utilizados. Algumas dessas etapas consistem no corte, costura e colocação de material aderente nas solas de forma ao calçado ser mais confortável.

2.1.1 *Welt Sewing Machine*

A *Welt Sewing* [3], figura 2.1, é uma máquina que faz a costura das palmilhas interiores ao sapato, para que o conforto e a resistência do calçado sejam o maior possível.

Em termos de segurança a *Welt Sewing* tem um pedal que dita o funcionamento da máquina. Caso o pedal esteja a ser pressionado a máquina funciona, caso contrário a alimentação, da ferramenta, é cortada e esta deixa de funcionar.

Figura 2.1 - Máquina de utilizada na costura de calçado (Welt Sewing Machine)

2.1.2 *Thermoadesive Attaching Machine*

A máquina *Thermoadesive attaching* [4], figura 2.2, serve para a colocação de adesivo nas palmilhas interiores do calçado. O adesivo é aquecido pela máquina através de ar comprimido para que a colagem, com a sola, seja a mais forte e durável possível.

A máquina está, também, equipada com um pedal que corta o adesivo depois de finalizada a tarefa.

Este método é muito utilizado, pois facilita o processo de costura explicado em 2.1.1, tornando-o muito mais rápido.

Figura 2.2 - Máquina de colocar adesivo nas solas (Thermoadesive Attaching Machine)

2.1.3 *Inseam Trimming Machine*

A *Inseam Trimming* [5], figura 2.2, é uma máquina de corte que serve para aparar o calçado em fase de produção.

Depois da costura do revestimento às palmilhas interiores dos sapatos é normal existir algum material do revestimento em excesso, com a ajuda desta máquina é possível a remoção de forma limpa, cuidada e eficiente do material em demasia.

Figura 2.3 - Máquina de corte (Inseam Trimming Machine)

2.2 Robótica no fabrico de calçado

Todas as máquinas descritas anteriormente têm um problema em comum, a necessidade de ter um operário constantemente junto da respectiva máquina.

Com a introdução da robótica na indústria do calçado torna-se desnecessário a utilização de um operário para realizar a mesma tarefa que o robot, ficando este livre para executar outras funções. Para além de optimizar o trabalho dos operários dentro da empresa é, também, possível um melhoramento do calçado fabricado.

Hoje em dia é possível encontrar empresas de calçado que utilizam robots para realizar as tarefas descritas em 2.1. A Lirel é uma empresa metalomecânica, que se dedica ao desenvolvimento de sistemas logísticos e máquinas para a indústria do calçado [6].

Figura 2.4 - Braço robótico utilizado no transporte de calçado para a máquina [7]

Na figura 2.4 pode-se verificar que o robot funciona como um operário da máquina, ou seja, o robot coloca o sapato na máquina e esta executa a tarefa. Neste caso a máquina efectua testes de resistência ao calçado [7].

Figura 2.5 - Braço robótico a executar tarefa numa sola de sapato [7]

Na figura 2.5, ao contrário da 2.4, o robot funciona como um executor da tarefa. Como se pode verificar o robot está devidamente isolado não impondo qualquer tipo de perigo para o resto dos operários da empresa. Neste caso o robot efectua um processo de colocação de cola na sola de sapato [7].

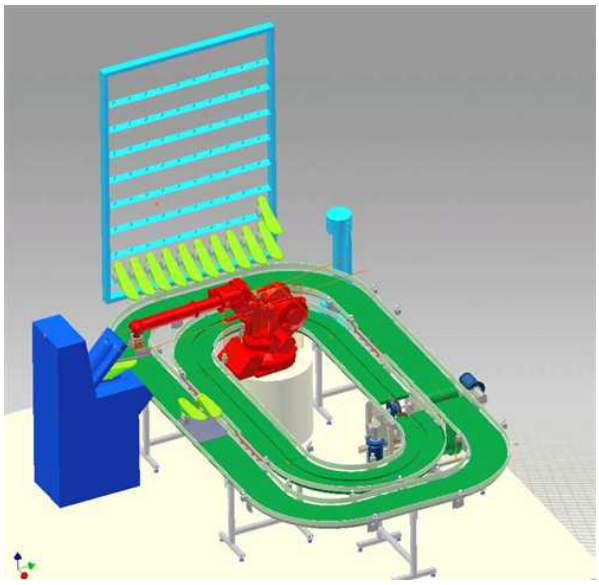


Figura 2.6 - Simulador utilizado pela Lirel [7]

Antes de colocar o robot a executar tarefas é normal a utilização de simuladores, não só para fazer estudos em termos da viabilização do sistema a implementar (Figura 2.6), mas também para serem efectuados testes de implementação através da sua programação off-line.

2.3 Visão por computador no fabrico de calçado

O controlo de qualidade na produção de qualquer tipo de produto é fulcral para o crescimento de qualquer empresa. Visto isto, as constantes inspecções e testes aplicados aos produtos prestam uma enorme contribuição para o seu desenvolvimento.

Uma das principais aplicações da visão por computador no fabrico de calçado é a inspecção. Através da visão por computador é possível encontrar falhas impossíveis de detectar ao olho humano nos produtos fabricados, tais como imperfeições suficientemente pequenas, no seu revestimento.

No fabrico de calçado a visão por computador é, também, utilizada na detecção do contorno da matéria-prima, para os sistemas de corte CAD/CAM poderem produzir o revestimento do calçado [8].

Este tipo de soluções de automação teve uma boa aceitação pelos fabricantes de calçado, pois é uma solução de custo relativamente baixo, boa resolução e pouco tempo de processamento [8].

Uma outra aplicação de visão por computador na indústria do calçado é um robot de desbaste de matéria-prima em excesso, que utiliza visão por computador em 3D para localizar as zonas do sapato onde tem de actuar [9].

2.4 Conclusões

Neste capítulo foram apresentados alguns dos mais importantes métodos de produção de calçado, aplicações da robótica na indústria do calçado e algumas utilizações da visão por computador no fabrico do mesmo.

O fabrico de calçado implica a realização de vários processos. Antigamente, estes eram realizados de forma manual, deixando demasiadas imperfeições e consequentemente produzindo um produto final de fraca qualidade. Hoje em dia, a produção de calçado melhorou a sua qualidade de produto graças à introdução de maquinaria na produção. No entanto, as máquinas de fabrico de calçado actuais têm, em comum o mesmo problema, a necessidade de uma pessoa operar a máquina. Com a introdução da robótica na indústria deixou-se de ter o problema supramencionado, pois o robot pode realizar todas as tarefas executadas por essas máquinas.

A visão por computador tem um papel relevante na indústria do calçado, pois possibilita ao consumidor um produto com muito mais qualidade, visto que é utilizada, na maior parte dos casos, em controlo de qualidade.

Capítulo 3

3 Sistema de visão por computador

O processamento de imagem consiste na caracterização e interpretação da informação contida em imagens obtidas no mundo real [10]. Devido aos grandes avanços industriais, cada vez mais, este tipo de inovação tornou-se convenientemente alvo de muitos estudos. Através de investigações foi possível ter uma melhor percepção no uso, essencial, destes sistemas nas diferentes áreas industriais.

Neste capítulo são inicialmente apresentadas algumas das técnicas de iluminação mais usuais em processamento de imagem. De seguida, são explicados algumas das técnicas de visão por computador utilizadas nesta dissertação, os seus algoritmos e alguns resultados obtidos.

3.1 Iluminação

A iluminação dos objectos é um factor muito importante na visão por computador. Uma má iluminação faz com que a aquisição da imagem seja feita de forma defeituosa, o que leva a uma difícil interpretação da mesma e impossibilita a realização e aplicação correcta de alguns algoritmos de processamento de imagem.

No entanto, existem algumas técnicas de iluminação que podem suportar a execução do projecto em questão. Entre as mais importantes estão a iluminação frontal e retro iluminação, figura 3.1 (a) e figura 3.1 (b), respectivamente.

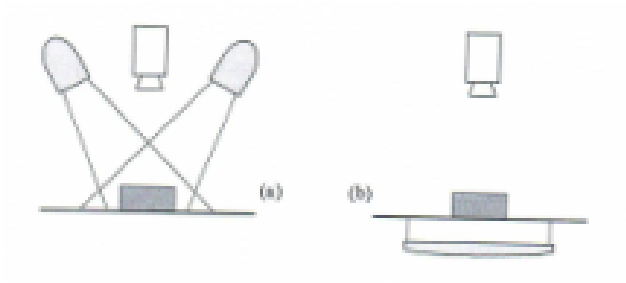


Figura 3.1 - Técnicas de iluminação; (a) - Iluminação frontal; (b) - Retro Iluminação

A técnica de iluminação frontal tem como objectivo iluminar a superfície do objecto, realçando todas as suas características, como a sua forma e dimensões. As fontes luminosas estão localizadas em cima do objecto e direccionadas para a sua superfície (Figura 3.1 (a)).

A técnica de retro iluminação tem como objectivo realçar os contornos dos objectos. A fonte luminosa, neste caso, encontra-se por trás do objecto, um dos exemplos desta técnica são as mesas de luz ou mesas iluminadas, muito utilizadas na medicina para visualização de radiografias. Os objectos são colocados nestas mesas, estando uma câmara por cima deste e apontando para esta (figura 3.1 (b)).

À primeira vista a técnica de retro iluminação parece a mais adequada para este projecto. No entanto, uma das preocupações desta dissertação é a utilização de soluções economicamente viáveis e como a iluminação frontal revelou-se eficaz, optou-se pela utilização da mesma.

Pode-se constatar, através das figuras 3.2 e 3.3, que sem a utilização deste tipo de equipamento seria impossível realizar, com sucesso, este projecto.

Figura 3.2 - Imagem do calcanhar da sola sem iluminação

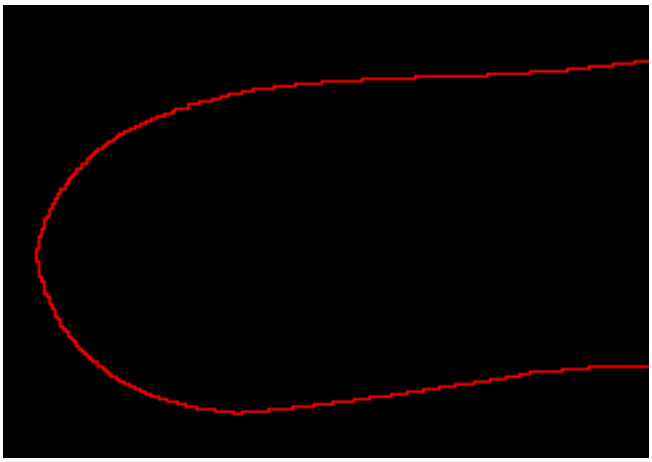


Figura 3.3 - Imagem do calcanhar da sola com iluminação

Pela observação das figuras anteriores, pode-se concluir que a 3.2 demonstra grandes imperfeições na sua aquisição, o que levaria à obtenção errada de resultados de eventuais algoritmos aplicados. Ao contrário, a figura 3.3 mostra uma imagem sem quaisquer defeitos.

3.2 Câmara CCD Firewire DFK21F04

Na aquisição de imagem foi utilizada uma câmara Firewire da ImagingSource, modelo DFK21F04 conforme é mostrada na figura 3.5. Esta câmara utiliza a tecnologia CCD (*charged-coupled device)*.

A tecnologia CCD (figura 3.4) não é mais do que um transdutor que transforma energia luminosa em energia eléctrica, sendo a quantidade de luz recebida proporcional à quantidade de carga eléctrica produzida. A energia eléctrica gerada é armazenada em pequenas células que correspondem aos pixéis da imagem [11].

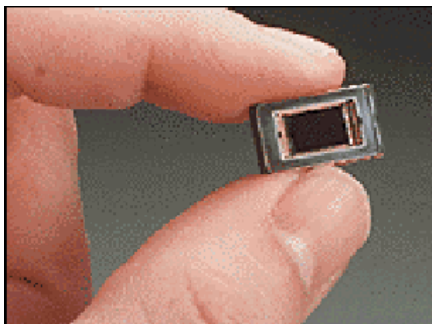


Figura 3.4 - Exemplo de um CCD [11]

Figura 3.5 - Câmara DFK21F04

As principais características desta câmara são:

- Sensor Sony CCD
- 640×480 Pixéis
- 60 Imagens/s

Através das tabelas 3.1, 3.2 e 3.3 podem-se verificar a performance, interface eléctrico, dimensões e peso da máquina.

Tabela 3.1 - Performance

Formatos de vídeo	640x480 UYVY @ 30, 15, 7.5, 3.75 fps 640x480 BY8 @ 60, 30, 15, 7.5, 3.75 fps
Sensibilidade	0.10 lx
DR	8 bit
SNR	ADC: 9 bit at 25°C, gain 0 dB

Tabela 3.2 - Interface eléctrico

Alimentação	8 a 30 VDC
Consumo de corrente	200 mA a 12 VDC

Tabela 3.3 - Dimensões e peso

Dimensões	H: 50.6 mm, W: 50.6 mm, L: 50 mm
Peso	265 g

3.3 Técnicas de processamento de imagem

A figura 3.6 apresenta o diagrama funcional do sistema de visão implementado.

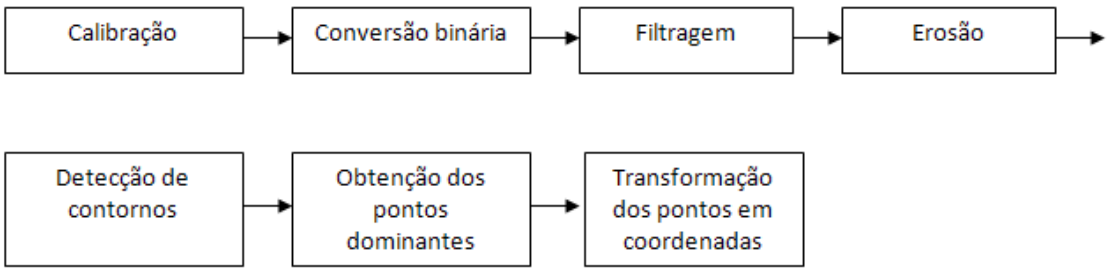


Figura 3.6 - Diagrama funcional das técnicas de visão implementadas

3.3.1 Calibração

A imagem captada pela câmara e transmitida para o PC não corresponde fielmente à imagem real, devido à distorção provocada pela lente.

A distorção por vezes acontece devido à própria natureza da luz ou fabrico defeituoso das lentes utilizadas nas câmaras [12]. A distorção das lentes pode ser para o exterior ou para o interior conforme é mostrado na Figura 3.7.

Figura 3.7 - Distorção para exterior (à esquerda) e para o interior (à direita) [13]

A calibração tem por objectivo compensar a distorção, tipicamente esta é feita pela aplicação de equações matemáticas ou pela restrição da zona de trabalho da imagem. Tornando, assim, os erros de distorção suficientemente pequenos de forma a poderem ser ignorados.

Pode-se entender, também, por calibração a conversão dos pixéis da imagem em coordenadas do mundo real, com a finalidade de se saber qual a dimensão dos objectos na realidade. Para tal será necessário saber a dimensão, em unidades SI, de um pixel. Assim sendo, é necessário encontrar-se uma solução onde através das coordenadas pixel (x_{imagem}, y_{imagem}) se obtêm as coordenadas reais (x_{real}, y_{real}).

Na lente da câmara passam todos os raios de luz que formam a imagem, deste modo é lógico afirmar que a cada raio de luz corresponde um pixel da imagem, ou seja, para cada coordenada pixel há-de corresponder uma coordenada física (Figura 3.8).

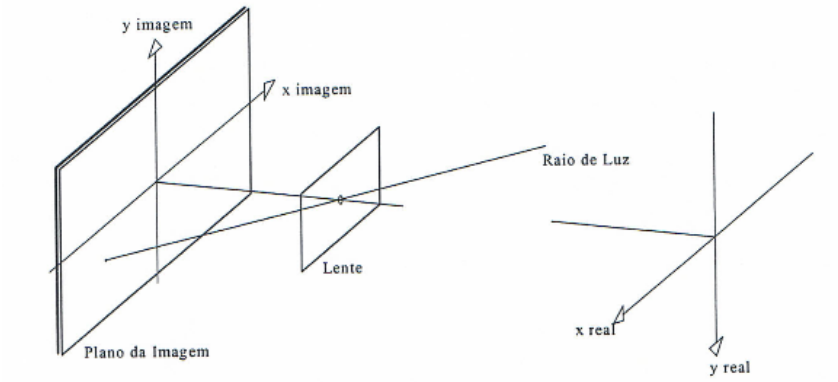


Figura 3.8 - Calibração da câmara

Visto que $\Delta x_{real} = x_{realmax} - x_{realmin}$ ou $\Delta y_{real} = y_{realmax} - y_{realmin}$ é directamente proporcional a $\Delta x_{imagem} = x_{imagemax} - x_{imagemin}$ e $\Delta y_{imagem} = y_{imagemax} - y_{imagemin}$, respectivamente, então podem-se relacionar as duas grandezas, calculando assim a dimensão física (D_{pixel}) de cada pixel da imagem.

Como a cada pixel irá corresponder um D_{pixel} então a cada Δx_{imagem} irá corresponder um $\Delta x_{real,}$ assim como a um Δy_{imagem} corresponde um Δy_{real}, pode-se concluir:

$$D_{pixel} = \frac{\Delta x_{real}}{\Delta x_{imagem}} \quad (3.1)$$

ou

$$D_{pixel} = \frac{\Delta y_{real}}{\Delta y_{imagem}} \quad (3.2)$$

Para se calcular os valores reais das coordenadas de cada pixel (x_{real}, y_{real}) faz-se o produto entre o D_{pixel} e as coordenadas pixel (x_{imagem}, y_{imagem}). Obtendo-se:

$$x_{real} = D_{pixel} \times x_{imagem} \quad (3.3)$$

e

$$y_{real} = D_{pixel} \times y_{imagem} \quad (3.4)$$

Achando, assim, as coordenadas físicas de cada ponto através das coordenadas pixel da imagem.

Neste projecto, não se achou pertinente a utilização de calibração para a distorção das lentes, pois, através da visualização das imagens não se encontrou qualquer tipo de razão que prove o contrário. No entanto, foi necessário calibrar a imagem relativamente à posição da câmara em relação ao objecto (Figura 3.9).

Através das equações (3.1) ou (3.2) é possível calcular as dimensões físicas do pixel, em centímetros.

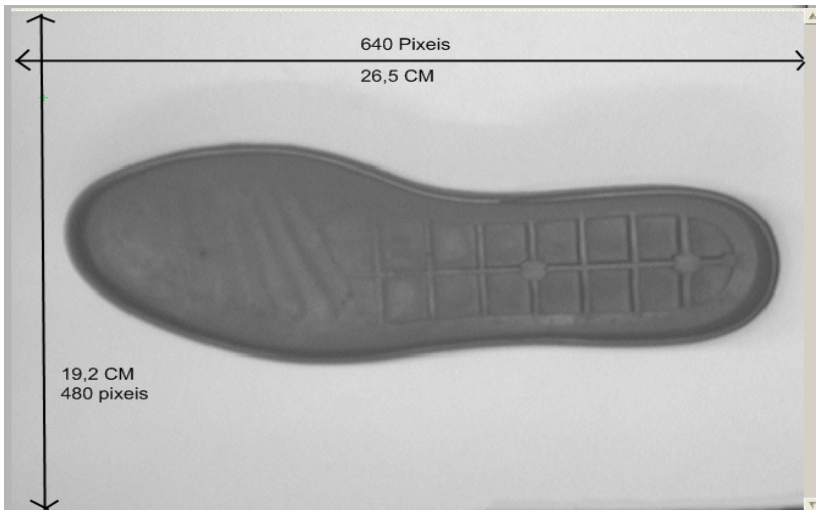


Figura 3.9 - Dimensões da imagem

$$D_{pixel} = 26,5 / 640 = 0,04 \text{ Centímetros}$$

$$D_{pixel} = 19,2 / 480 = 0,04 \text{ Centímetros}$$

Através do cálculo do pixel em ambas as orientações, é possível afirmar que todos os pixéis da imagem têm dimensões de 0,04 por 0,04 centímetros.

3.3.2 Conversão binária ou *Thresholding*

Um ser humano consegue, facilmente, visualizar uma imagem e distinguir o objecto do fundo da mesma. Contrariamente, um computador não o consegue fazer sem converter a imagem em binário. Assim sendo, esta técnica é um processo que divide a imagem em duas regiões distintas.

O processo de *Thresholding* é feito com a ajuda de um histograma. Um histograma é um gráfico que mostra a distribuição do número de pixéis em função da sua intensidade, sendo a sua intensidade dada em tons de cinzento [14] (Figura 3.10).

Figura 3.10 - Intensidade dos pixéis em tons de cinzento (0 - preto até 255 - branco) [14]

Através do histograma é possível fazer a distinção entre o objecto e o fundo, figura 3.11.

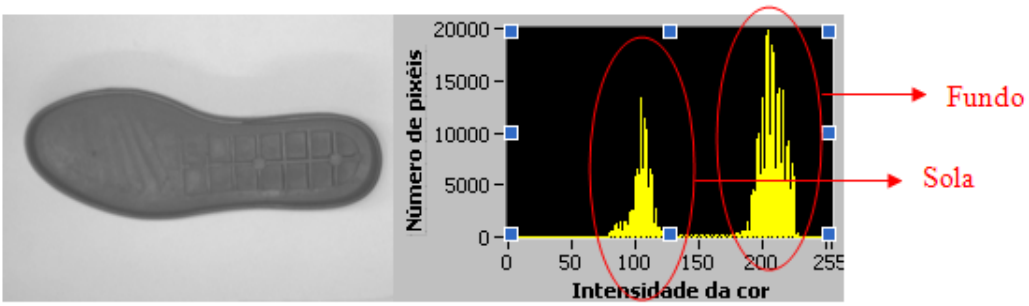


Figura 3.11 - Imagem da sola e respectivo histograma

Como se pode verificar na figura 3.11, o histograma mostra que na imagem a maior parte dos pixéis estão próximo de 255 (branco) que corresponde ao fundo, com um tom mais claro, e que existe também uma quantidade mais pequena de pixéis mais próximo de 0 (preto) que corresponde à sola, com um tom mais escuro.

Para além do histograma informar acerca da quantidade de pixéis em função do seu tom é possível, também, extrair o valor de *thresholding*. A partir da figura 3.11 pode verificar-se que esse valor será aproximadamente 150.

O software fará a que todos os pixéis que estiverem abaixo de 150 (sola) seja atribuído o nível lógico 0, e todos os que fiquem acima (fundo) seja atribuído o nível lógico 1. Assim, é viável converter qualquer imagem em binário, sendo possível distinguir o objecto do fundo desde que haja contraste suficiente na imagem.

Matematicamente, o *Thresholding* pode-se traduzir da seguinte forma;

Sendo uma imagem $f(x, y)$ convertida numa imagem binária $g(x, y)$, obtêm-se a seguinte função:

$$g(x,y) = \begin{cases} 0 \; se \; f(x,y) \leq T \\ 1 \; se \; f(x,y) > T \end{cases} \quad (3.5)$$

Através desta função conclui-se que qualquer pixel que estiver abaixo do nível de *thresholding* (*T*) é considerado nível lógico 0, caso esteja acima considera-se nível lógico 1 (Figura 3.12).

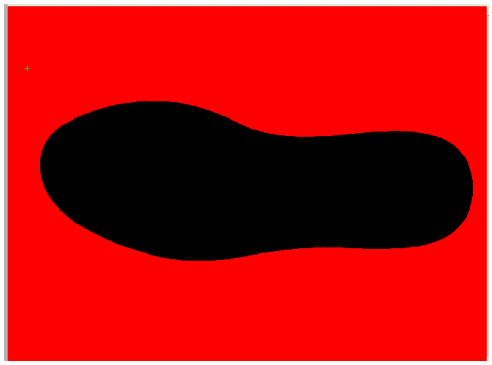


Figura 3.12 - Imagem binária

3.3.3 Filtragem

A aplicação de filtros em processamento de imagem é muito usual. Com a utilização de filtros é possível, por exemplo: a redução de ruído na imagem, a detecção de contornos, a extracção de zonas de interesse, entre outros. Um filtro não é mais do que uma matriz bidimensional (denominada de máscara), geralmente 3x3 ou 5x5, que percorre toda a matriz da imagem e opera sobre os pixéis da mesma. A estes filtros são dados o nome de espaciais [15].

O filtro utilizado nesta etapa é um passa-baixo com uma mascara 3x3, e serve para suavizar os contornos e realçar os detalhes da imagem adquirida. Como foi dito anteriormente, o filtro irá operar sobre os pixéis da matriz da imagem, percorrendo e substituindo cada um deles, pela máscara utilizada.

Os mecanismos deste tipo de filtro consistem no movimento do ponto central da máscara por todos os pixéis da imagem. A resposta da máscara, a cada ponto da

imagem, irá ser a soma dos produtos dos coeficientes do filtro com os coeficientes da imagem (Figura 3.13) [15].

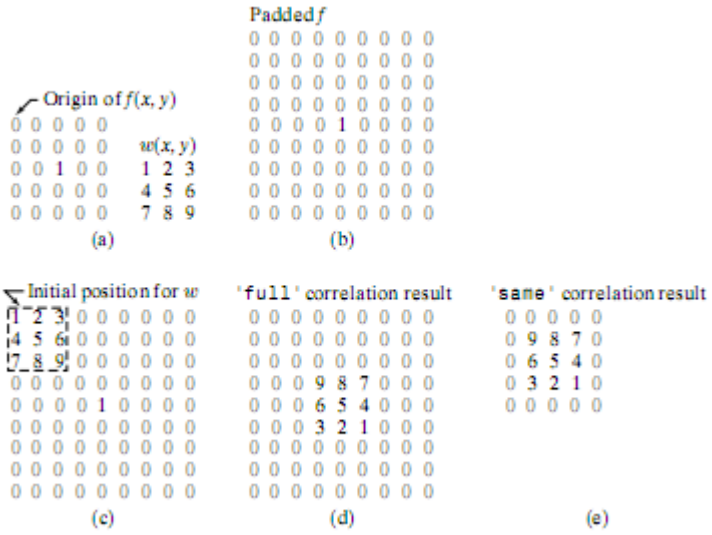


Figura 3.13 - Exemplo de filtragem de uma imagem [15]

Na figura 3.13 (a) encontra-se uma imagem $f(x, y)$ com um pixel a nível lógico 1 e os restantes a nível lógico 0. Além da imagem, existe também, um filtro $w(x, y)$. No exemplo (c) a máscara coloca-se na posição inicial da imagem, enquanto os coeficientes da imagem forem zero, o filtro não tem qualquer influência sobre a mesma, pois a soma de produtos de ambos é zero. A interferência do filtro só se fará notar nas imediações do pixel a nível lógico 1 da imagem. Centrando o filtro no elemento $f(4,4)$ da imagem, consegue-se obter o primeiro elemento da imagem filtrada, sendo este o valor 9 calculado através da multiplicação do valor 9 do filtro $w(x, y)$ com o valor 1 da imagem $f(x, y)$. Movendo o filtro para a posição $f(4,5)$ irá ser obtido o valor 8 utilizando o método acima descrito. Este processo é realizado até à imagem ser totalmente percorrida. No final de todas as movimentações do filtro $w(x, y)$ obtém-se a figura 3.13 (d) totalmente filtrada.

Na figura 3.14 pode-se verificar o efeito do filtro aplicado. Na imagem original, à esquerda, notam-se algumas imperfeições nos contornos do objecto, com a aplicação do filtro, à direita, essas imperfeições são facilmente corrigidas.

Figura 3.14 - Imagem original da sola à esquerda e imagem filtrada à direita

3.3.4 Erosão

O processo de halogenação utiliza-se no contorno interior das solas, mas a imagem adquirida, pela câmara, mostra como sendo o contorno no bordo das mesmas. Como o desejável será aplicar o químico no contorno interior, utiliza-se a técnica de erosão. Ao erodir a imagem é possível fazer com que os limites fiquem dentro das solas conforme o pretendido.

Tecnicamente, a erosão pode ser vista como uma remoção de toda uma região de interesse de uma imagem. Para isso, é necessária a utilização de uma máscara 3x3, 5x5 ou 11x11 que percorrerá a imagem erodindo-a conforme as combinações dos seus coeficientes (Figura 3.15).

Figura 3.15 - Exemplo de erosão de um objecto circular e respectivas matrizes: (a) imagem original; (b) imagem erodida na orientação pretendida; (c) imagem erodida na orientação pretendida.

Na figura 3.15, é possível concluir que a imagem é erodida consoante a matriz introduzida. A colocação dos coeficientes a nível lógico alto na matriz de erosão irá fornecer a orientação de erosão. No caso desta aplicação concreta foi aplicada uma máscara 3x3, todos os seus coeficientes têm que estar a nível lógico alto, figura 3.16, pois a erosão será feita em toda a imagem.

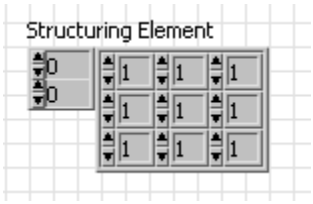


Figura 3.16 - Matriz de erosão

A figura 3.17 mostra a imagem original e a imagem após a aplicação da técnica de erosão.

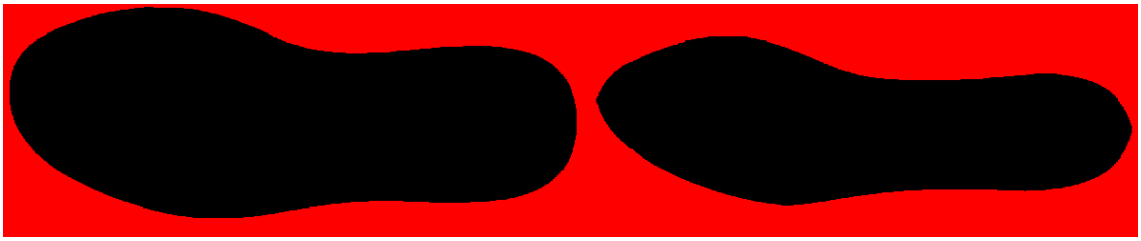


Figura 3.17 - Imagem original à esquerda e imagem erodida à direita

3.3.5 Detecção de contornos

A detecção de contornos é das ferramentas mais utilizadas em processamento de imagem, pois, permite reduzir drasticamente a quantidade de informação a ser processada, em cada imagem. Ao mesmo tempo, possibilita um conhecimento muito preciso acerca dos limites do objecto observado [16].

Numa imagem estão, geralmente, associadas descontinuidades entre o fundo e o objecto, essas diferenças podem ser detectadas através de filtros. Esses filtros irão realçar os contornos do objecto, tornando a imagem muito mais perceptível [15], sendo designados por detectores de contorno.

Entre os mais conhecidos detectores de contorno, estão os filtros de Sobel, de Prewitt e de Roberts. Os operadores de Sobel e Prewitt são formados por duas máscaras 3x3 cada um, uma na orientação x, e outra na orientação y que percorrem toda a imagem, calculando o gradiente para cada pixel, em cada uma das direcções. Enquanto o de Roberts apenas usa duas máscaras 2x2. As máscaras utilizadas pelo operador Sobel são:

$$Gx = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix} e \; Gy = \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix}.$$

Na filtragem de Prewitt as máscaras utilizadas são:

$$Gx = \begin{bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix} e \; Gy = \begin{bmatrix} -1 & -1 & -1 \\ 0 & 0 & 0 \\ 1 & 1 & 1 \end{bmatrix}.$$

O filtro detector de contornos Roberts utiliza as máscaras 2x2:

$$Gx = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} e \; Gy = \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix}.$$

As máscaras de Prewitt, Sobel e Roberts percorrem a imagem utilizando os mesmos princípios que os filtros de suavização explicados anteriormente, contudo utilizam as suas máscaras para o cálculo do gradiente em ambas as orientações [17].

Na visão por computador, a caracterização dos contornos de um objecto é dado por descontinuidades nas intensidades dos pixéis de uma imagem $f(x, y)$. Para a detecção destas descontinuidades é feito o cálculo do gradiente. Basicamente, o gradiente é a alteração do valor de uma quantidade por unidade de espaço, logo esta é a ferramenta ideal para a detecção das variações de intensidades dos pixéis. Matematicamente, o gradiente, é dado por:

$$\|\nabla f(x, y)\| = \begin{bmatrix} Gx \\ Gy \end{bmatrix} = \begin{bmatrix} \frac{\partial f(x,y)}{\partial x} \\ \frac{\partial f(x,y)}{\partial y} \end{bmatrix}. \quad (3.6)$$

Através do cálculo das derivadas de 1ª ordem de $f(x, y)$ obtém-se a variação de f no ponto (x, y). Para a determinação do contorno é imprescindível obter a magnitude do gradiente a cada ponto da imagem $f(x, y)$ sendo calculado por:

$$\|\nabla f(x,y)\| = \sqrt{Gx^2 + Gy^2}. \quad (3.7)$$

Para o cálculo da magnitude do gradiente são necessárias ambas as orientações Gx e Gy. Com a utilização, por exemplo, do operador Sobel, $Gx = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix}$ e $Gy = \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix}$, numa imagem $f(x,y)$ é possível obter-se ambas as orientações:

$$Gx = \left(f(x_{i-1}, y_{j+1}) + 2f(x_i, y_{j+1}) + f(x_{i+1}, y_{j+1})\right) - \left(f(x_{i-1}, y_{j-1}) + 2f(x_i, y_{j-1}) + f(x_{i+1}, y_{j-1})\right) \quad (3.8)$$

$$Gy = \left(f(x_{i+1}, y_{j-1}) + 2f(x_{i+1}, y_j) + f(x_{i+1}, y_{j+1})\right) - \left(f(x_{i-1}, y_{j-1}) + 2f(x_{i-1}, y_j) + f(x_{i-1}, y_{j+1})\right) \quad (3.9)$$

Após o cálculo da magnitude do gradiente, para cada pixel, será indispensável a comparação desse valor com um limite T, para o qual será considerado ponto de contorno ou não. Matematicamente pode ser visto pela seguinte função:

$$g(x,y) = \begin{cases} 0 \; se \; \nabla f(x,y) \leq T \\ 1 \; se \; \nabla f(x,y) > T \end{cases} \quad (3.10)$$

Caso o gradiente do pixel (x,y) da imagem f for maior que o limite T, então esse pixel fica a nível lógico 1, caso contrário fica a nível lógico 0.

Todos os filtros detectores de contorno mencionados anteriormente, utilizam o mesmo método previamente explicado, tendo como única diferença a máscara utilizada.

Obviamente, a sua utilização deve ser feita consoante a aplicação, pois cada um dos filtros funciona melhor conforme o tipo de objecto adquirido na imagem (Figura 3.18).

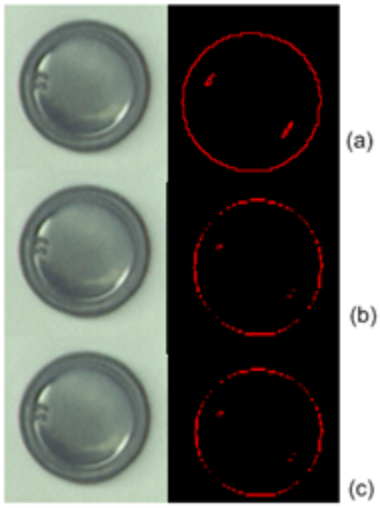


Figura 3.18 - Exemplo de aplicação de filtros de detecção de contorno; (a) - Roberts; (b) - Sobel; (c) - Prewitt

Através da figura 3.18, é possível concluir que, para este objecto, o filtro Roberts é o mais adequado. Os filtros Prewitt e Sobel deixam demasiadas imperfeições na imagem do objecto.

Relativamente à detecção de contorno das solas tem-se a figura 3.19.

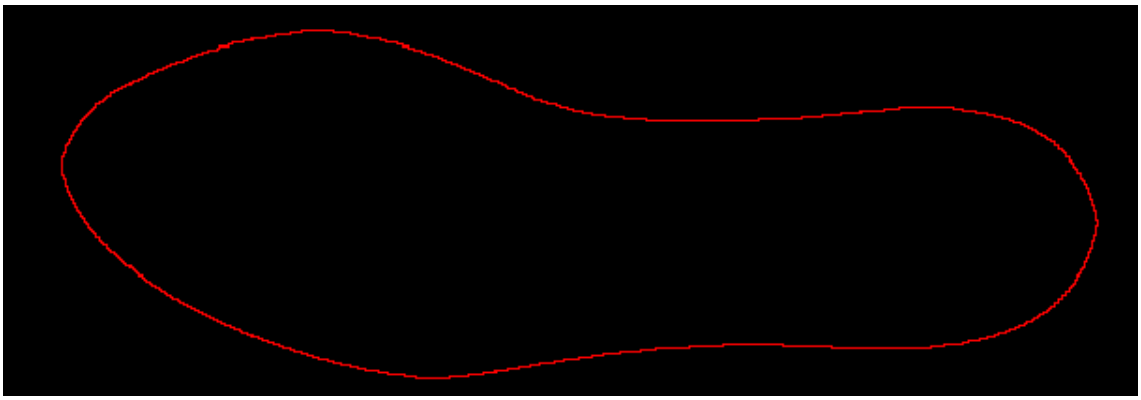


Figura 3.19 - Contorno da sola

3.3.6 Obtenção dos pontos dominantes

A detecção de contornos, explicada anteriormente, possibilita uma redução significativa da quantidade de informação a ser processada, pois o objecto fica reduzido apenas ao seu contorno.

Com a obtenção dos pontos dominantes, é permissível uma redução ainda maior da quantidade de dados a ser processada, pois, com uma menor quantidade de pontos é possível definir o mesmo objecto. Recorde-se, que é necessário definir uma zona de trabalho para o robot, ou seja, pretende-se um conjunto de pontos/coordenadas que definam a sua região de trabalho. Portanto, descrição pixel a pixel fornecida pela detecção de contornos torna-se pouco eficiente, pois o número de pixéis que definem a totalidade do objecto é, geralmente grande. Assim sendo, a extracção dos pontos dominantes são o instrumento ideal para este tipo de problema. Os pontos dominantes podem ser vistos como um menor conjunto de pixéis, relativamente aos de contorno, que permitem a representação do objecto.

Os algoritmos de obtenção dos pontos dominantes têm que ter em atenção as concavidades mais acentuadas dos objectos, ou seja, caso o contorno de um objecto não tenha variações de curvatura, é apenas necessária a marcação de um número reduzido de pontos (Figura 3.20 (a)). Caso o objecto tenha concavidades demasiadamente acentuadas é fundamental a marcação de um maior número de pontos ao longo da sua curvatura (Figura 3.20 (b)).

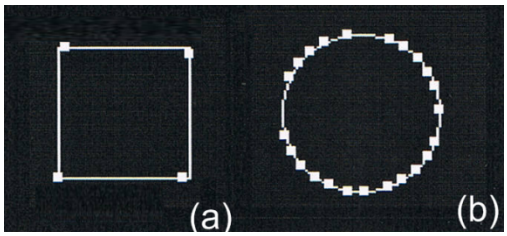


Figura 3.20 - Exemplo de extracção de pontos dominantes de diferentes objectos; (a) - Quadrado; (b) - Circulo

3.3.6.1 Algoritmo de aproximação à tangente

Entre os algoritmos mais utilizados da obtenção dos pontos dominantes está o método de aproximação à tangente, onde são calculadas as tangentes de dois pontos e utilizado o desvio angular entre elas como critério de decisão (Figura 3.21).

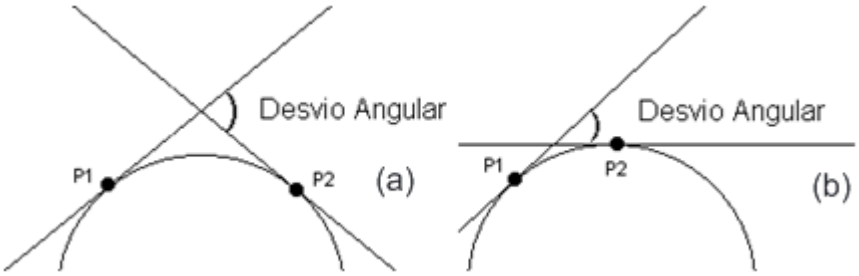


Figura 3.21 - Ilustração do algoritmo das tangentes; (a) - Espaçamento maior entre os pontos; (b) Espaçamento menor entre os pontos.

A partir da figura 3.21 pode-se verificar que o espaçamento entre os pontos relaciona-se com o desvio angular fornecido pelas suas tangentes, quanto maior o espaçamento, maior será o desvio angular. Logo, um bom critério de decisão será limitar o desvio angular a um valor máximo, sempre que se ultrapasse esse valor será, obrigatoriamente, marcado um ponto.

No exemplo da figura 3.21, uma boa selecção de pontos será a da imagem (b), pois como a curvatura é demasiado acentuada, sendo necessária uma marcação menos espaçada entre os pontos. Contrariamente, a imagem (a) é exemplo de uma má escolha de pontos, pois o espaçamento de pontos é demasiado grande tornando-se difícil definir o objecto com o maior rigor e exactidão possível. Para além de uma definição defeituosa do objecto, o robot não percorrerá com precisão o seu contorno.

3.3.6.2 Algoritmo de aproximação poligonal

Uma outra forma de obter os pontos dominantes é através do algoritmo de aproximação poligonal, onde a obtenção de pontos é feita através da marcação de uma recta entre o início e o fim do contorno. É esperado que o algoritmo meça a distância entre o segmento de recta e o contorno do objecto (Figura 3.22 (b)). Sempre que o valor da distância, pré-definido, seja atingido é marcado um ponto de contorno. Após a marcação é traçado um segmento de recta entre o ponto obtido e o final do contorno para a obtenção do seguinte ponto (Figura 3.22 (c)). O algoritmo repete-se até ao último ponto marcado coincidir com o ponto final do contorno [1].

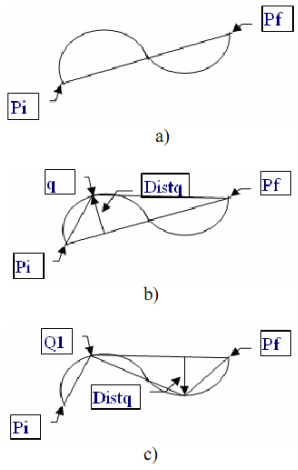


Figura 3.22 - Exemplo de algoritmo de aproximação poligonal; (a) - segmento de recta entre ponto inicial e final; (b) - marcação do primeiro ponto; (c) - marcação do segundo ponto [1]

3.3.6.3 Algoritmo utilizado

Apesar dos algoritmos explicados anteriormente serem eficazes, o algoritmo utilizado neste projecto foi diferente, por motivos de facilidade de implementação.

No LabView existe uma funcionalidade que permite saber qual a curvatura de pixéis consecutivos (Figura 3.23 (c)).

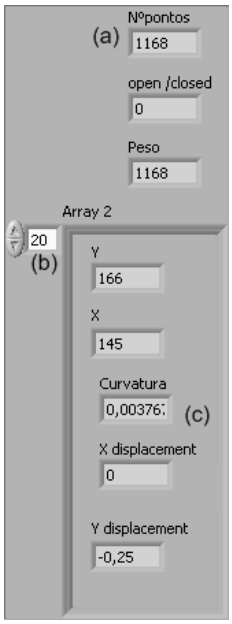


Figura 3.23 - (a) - Número de pixéis do contorno; (b) - Número de pixel actual; (c) - Curvatura do pixel actual em relação ao anterior.

Quanto maior for a curvatura entre pixéis, maior será a curvatura do objecto, logo um bom critério de decisão será a curvatura entre pixéis fornecida pelo software. Como a curvatura é dada entre o pixel actual e o anterior, torna-se impossível seleccionar qualquer ponto de contorno, devido ao curto intervalo entre pixéis. Para resolver este problema somou-se as curvaturas entre vários pixéis consecutivos, quando estes atingem um determinado valor acumulado, marca-se um ponto dominante. Nas curvaturas mais acentuadas da sola, o valor acumulado é atingido mais rapidamente, logo existirão mais pontos marcados, ao contrário dos locais com curvaturas menos salientes, onde se marcarão menos pontos.

Por exemplo utilizando um valor acumulado de 0,4 obtém-se a figura 3.24.

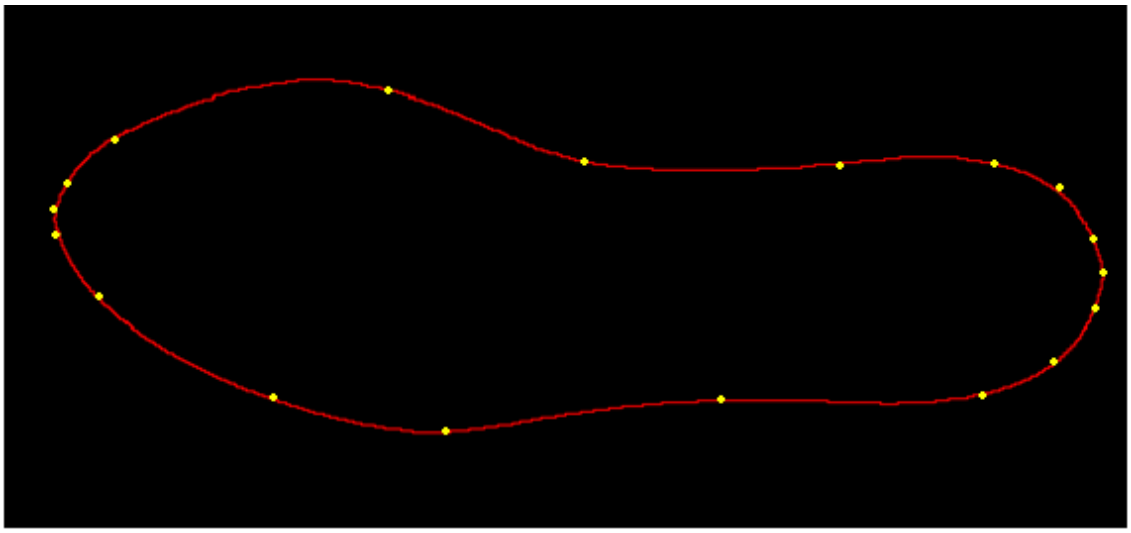


Figura 3.24 - Resultado dos pontos dominantes com valor máximo de 0,4

Utilizando o valor acumulado de 0,2 obtém-se mais pontos dominantes, Figura 3.25.

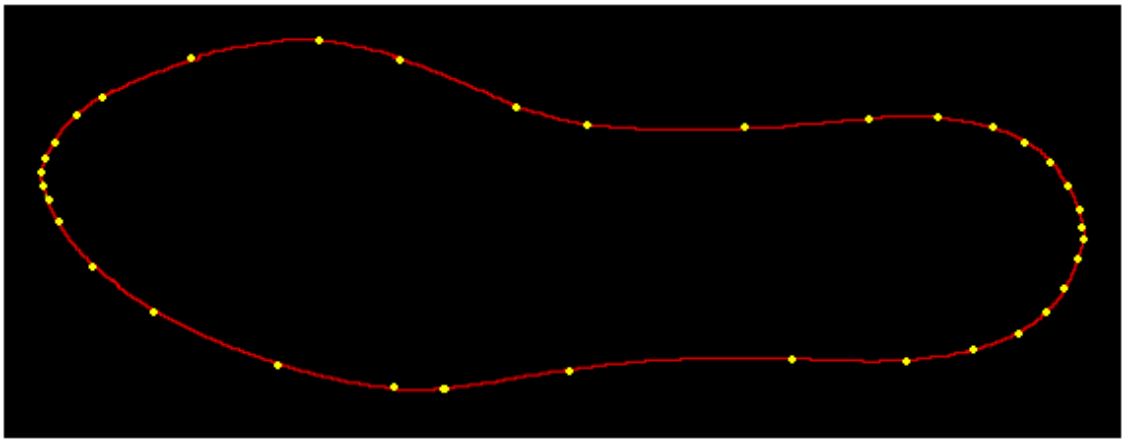


Figura 3.25 - Resultado dos pontos dominantes com valor máximo de 0,2

3.3.7 Transformação dos pontos em coordenadas reais

A obtenção dos pontos dominantes é muito importante neste projecto, pois serão estes os pontos que o robot irá percorrer para executar a tarefa de halogenação, no entanto os pontos retirados não tem uma correspondência directa com as coordenadas reais do mundo.

A aquisição dos pontos, é feita através da imagem digital do computador, ou seja as coordenadas obtidas são em relação à imagem (coordenadas pixel) e não ao mundo real (coordenadas reais). Para uma realização eficaz da tarefa o robot terá que percorrer os pontos dominantes reais do objecto.

A obtenção dos pontos reais, é feita com base na calibração da imagem, explicada em 3.3.1 Através das equações $x_{real} = D_{pixel} \times x_{imagem}$ e $y_{real} = D_{pixel} \times y_{imagem}$, é possível a transformação de todos os pontos obtidos em coordenadas do mundo real, prontas para serem transferidas para o robot.

Todos os pontos dominantes, obtidos anteriormente em 3.3.6.3, são guardados num *array* (Figura 3.26) à medida que são marcados no software. Para além da eficácia e ganho em tempo de execução, garante-se que os pontos são disponibilizados de forma ordenada.

Figura 3.26 - *Array* de coordenadas pixel dos pontos dominantes

Utilizando as coordenadas pixel da figura 3.26 e recordando que o valor de D_{pixel} é 0.04, calculado em 3.3.1, obtém-se:

$$x_{real} = 0{,}04 \times 147 = 5{,}88\ centímetros$$

$$y_{real} = 0{,}04 \times 234 = 9{,}36\ centímetros$$

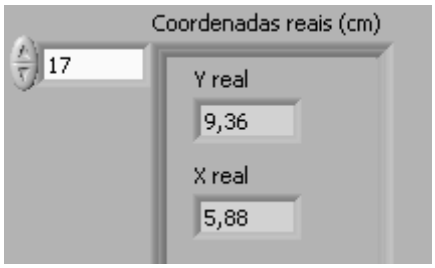


Figura 3.27 - Pontos dominantes reais

Comparando os valores obtidos com os da figura 3.27 verifica-se que estes são iguais.

3.4 Conclusões

Neste capítulo foram apresentados os conceitos teóricos de visão por computador e o problema de iluminação muita vezes encontrado neste tipo de projectos.

Com condições de iluminação muito deficientes torna-se impossível a utilização de sistemas de visão por computador, pois os resultados obtidos através dos algoritmos aplicados são errados. Visto isto, pode-se concluir que a iluminação é um dos factores mais importantes na utilização deste tipo de tecnologia.

Foram, também, aplicadas algumas técnicas de processamento de imagem que permitiram saber qual a dimensão dos pixéis (calibração), distinção entre o fundo e o objecto da imagem (conversão binária), alteração das propriedades do objecto (erosão), demonstração de alguns algoritmos de marcação de pontos de contorno e, finalmente, transformação dos pontos em coordenadas reais.

Capítulo 4

4 Aplicação em Labview

Para a implementação das técnicas visão por computador anteriormente descritas foi utilizada a plataforma de programação LabView da *National Instruments*. O LabView é uma plataforma de programação gráfica, que permite ao utilizador uma interacção amigável e de fácil compreensão com o programa em execução. Além disso, possibilita ao programador a utilização de algumas funções dedicadas a processamento de imagem, que facilitam a realização deste tipo de tarefas.

A programação em LabView utiliza o conceito de VI's ou *Virtual Instruments*, cada VI é formado por dois painéis: o primeiro é o painel frontal, onde estará o interface que será visto pelo utilizador, o segundo é o diagrama de blocos que contém o código do painel frontal (Figura 4.1). A programação deste tipo de plataforma é feita através de interligação de blocos, onde cada um deles corresponde a uma função. Assim, deixa-se de parte a escrita de várias linhas de código de linguagem complicada, que necessitam de programadores especializados e experientes.

Ao longo deste capítulo serão apresentados os códigos das técnicas implementadas e os protocolos de comunicação utilizados.

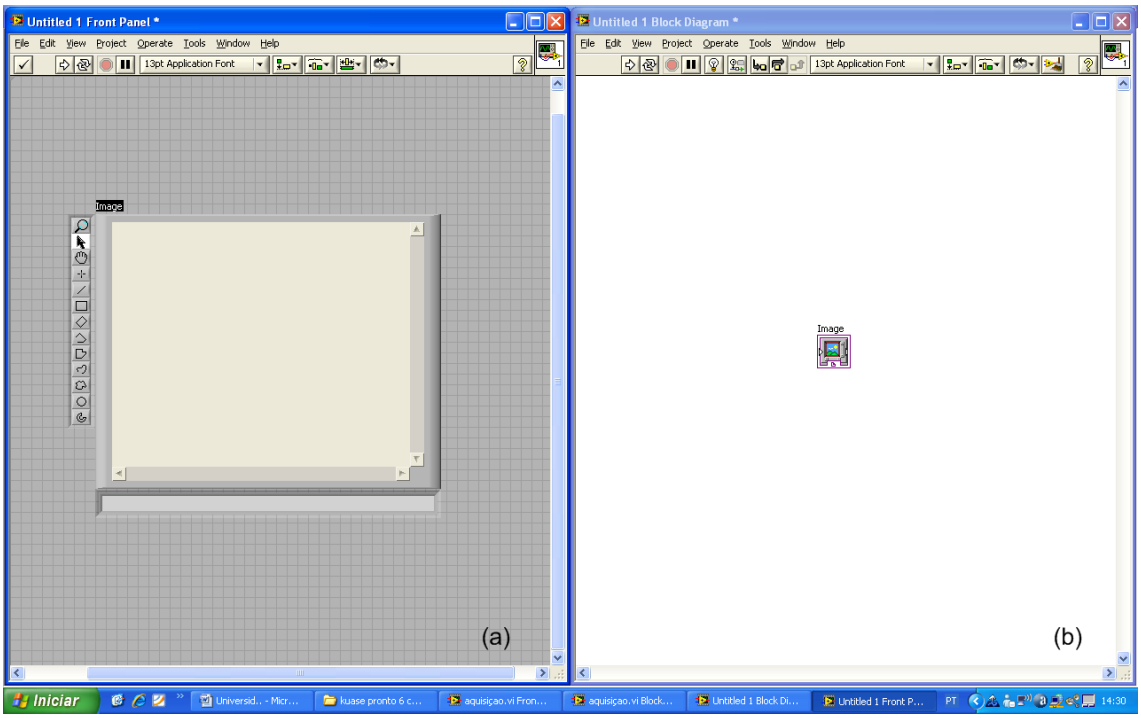


Figura 4.1 - Exemplo de interface LabView; (a) - Painel frontal; (b) - Diagrama de blocos

4.1 Código do sistema de visão por computador

Em termos de aquisição de imagem é fundamental a existência de alguns blocos do sistema de visão e movimento do LabView (Figura 4.3). É necessário, por exemplo, que o utilizador informe ao software qual o referência da câmara a adquirir (Figura 4.2).

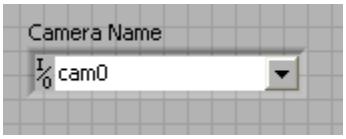


Figura 4.2 - Referência da câmara

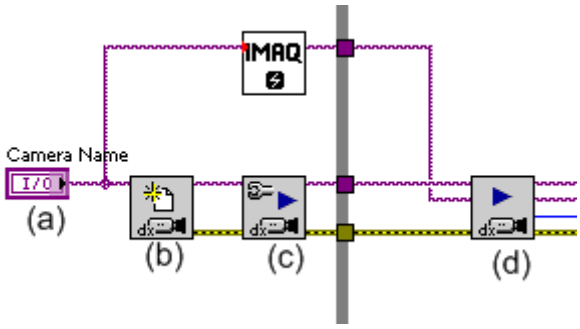


Figura 4.3 - Diagrama de blocos para adquirir a imagem

Basicamente, a referência da câmara (a) é enviada para ser configurada (b). No bloco (c) a imagem da câmara é aberta e disponibilizada frame a frame em (d). Finalmente o bloco (d) mostra o ultima frame recebida.

O código das técnicas de processamento de imagem explicadas no capítulo anterior tem-se na figura 4.4.

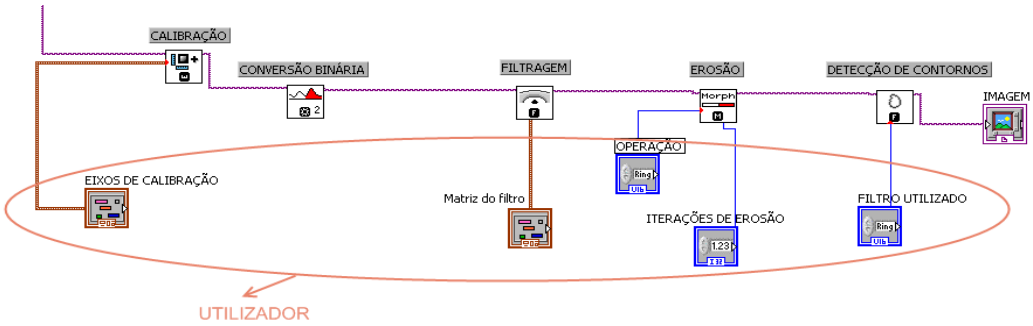


Figura 4.4 - Código das técnicas de processamento de imagem

Em termos de interface com utilizador, este apenas tem que introduzir o tamanho de cada pixel nos eixos de calibração (figura 4.5), a dimensão da matriz do filtro (figura 4.6), o tipo de operação e quantas iterações deseja na erosão (figura 4.7) e, finalmente, o filtro utilizado na detecção de contornos (figura 4.8).

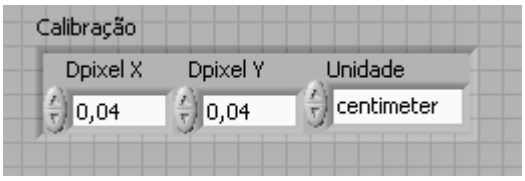


Figura 4.5 - Interface de calibração da imagem

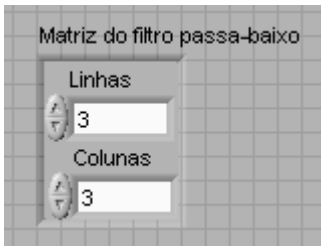


Figura 4.6 - Introdução da dimensão da matriz

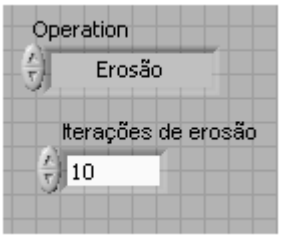


Figura 4.7 - Tipo de operação e número de iterações

Figura 4.8 - Filtro de detecção de contornos

Após a realização das técnicas de processamento de imagem é necessário implementar o algoritmo de marcação dos pontos dominantes (Figura 4.9).

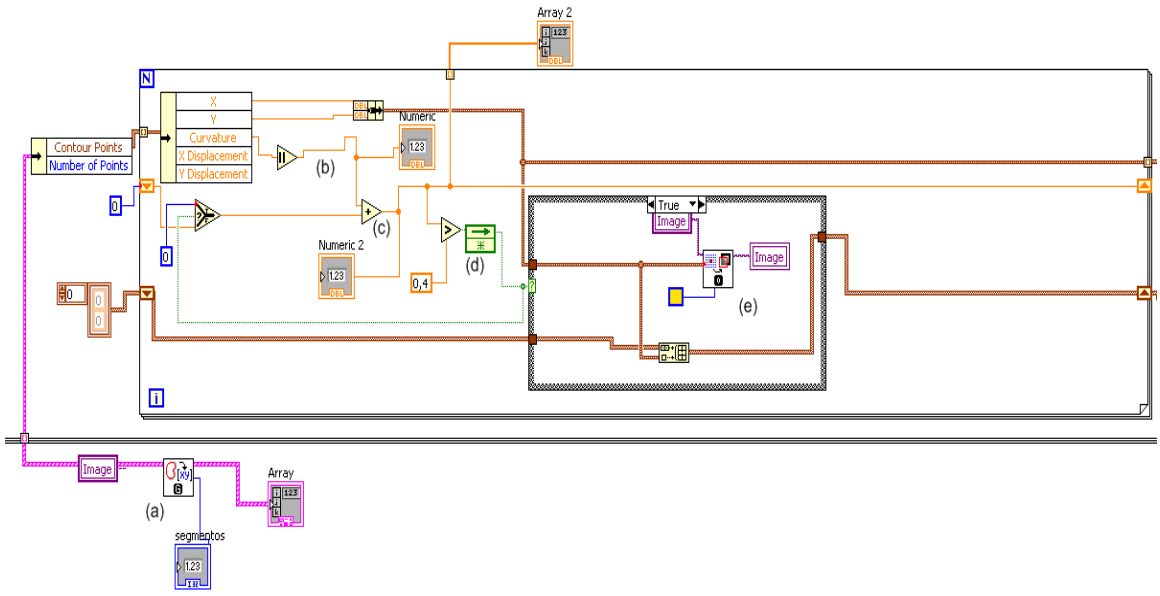


Figura 4.9 - Imagem de código da obtenção de pontos dominantes; (a) - Obtenção de todos os pontos de contorno em array; (b) - Disponibilização do valor da curvatura; (c) - Soma da curvatura com valor anterior; (d) - Teste ao valor acumulado; (e) – Marcação do ponto

A figura 4.9 mostra o código referente à marcação dos pontos dominantes na sola de sapato. A explicação deste pedaço de código é relativamente simples, em (a) obtém-se todos os pontos de contorno num *array*, assim como a curvatura entre cada pixel. Em (b) o valor da curvatura de um pixel inicial é disponibilizado para a soma com o valor da curvatura anterior, (c). Em (d) é feito o teste ao valor acumulado, caso este seja maior que 0,4, então marca-se um ponto dominante, (e). Sempre que o teste, (d), for falso o valor acumulado até então é disponibilizado para a próxima soma, (c), caso seja verdadeiro, é reiniciada a contagem, guardando as coordenadas do ponto marcado num *array*. Este procedimento repete-se até serem percorridas todos os pixéis de contorno da sola.

Para a transformação dos pontos dominantes em coordenadas do mundo real utiliza-se o bloco de software da figura 4.10, onde as coordenadas pixel são fornecidas e obtidas as coordenadas reais.

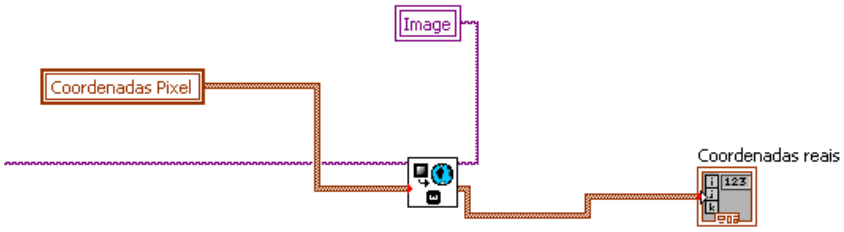


Figura 4.10 - Código da transformação dos pontos para o mundo real

4.2 Protocolo de comunicação

A comunicação com o robot é efectuada através do protocolo série RS-232. Para isso foi necessário definir um protocolo por software de envio dos pontos dominantes para o robot.

4.2.1 Configuração da porta série

Em termos de configuração da porta série utiliza-se um interface em LabView para definir o *Baud Rate*, *Data Bits*, *Parity*, *Stop Bits* (Figura 4.11).

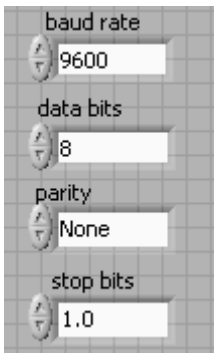


Figura 4.11 - Configuração da porta série

Relativamente ao código de configuração da porta série obtém-se a figura 4.12, onde as configurações introduzidas na figura 4.11 são enviadas para o bloco de configuração (a).

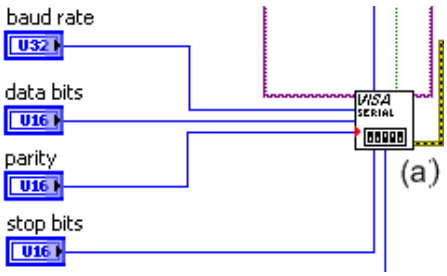


Figura 4.12 - Código da configuração da porta série; (a) - Bloco de abertura da porta série

4.2.2 Protocolo de envio

Após o armazenamento e transformação dos pontos dominantes em coordenadas do mundo real, é necessário o seu envio via porta série.

O envio é feito ponto a ponto, ou seja, a cada iteração é enviada a coordenada x e y de cada ponto dominante. Visto isto, é imprescindível a existência de um protocolo de comunicação entre o computador e o robot. Esse protocolo consistirá numa série de avisos, enviados, entre o computador e o robot, aquando das suas disponibilidades para ler ou escrever via porta série.

O protocolo utilizado consiste em o robot notificar o computador que está pronto para receber, e este envia o primeiro ponto. Quando o ponto é recepcionado com sucesso, o robot comunica ao computador novamente, informando que já recebeu o ponto. O ciclo é reiniciado para o envio da próxima coordenada. Após o envio de todos os pontos, o computador envia *strings* sem conteúdo. O robot deve ser capaz de identificar que está a receber informação inválida e notificar o computador para terminar a comunicação. O fluxograma da figura 4.13 exemplifica o processo anteriormente explicado.

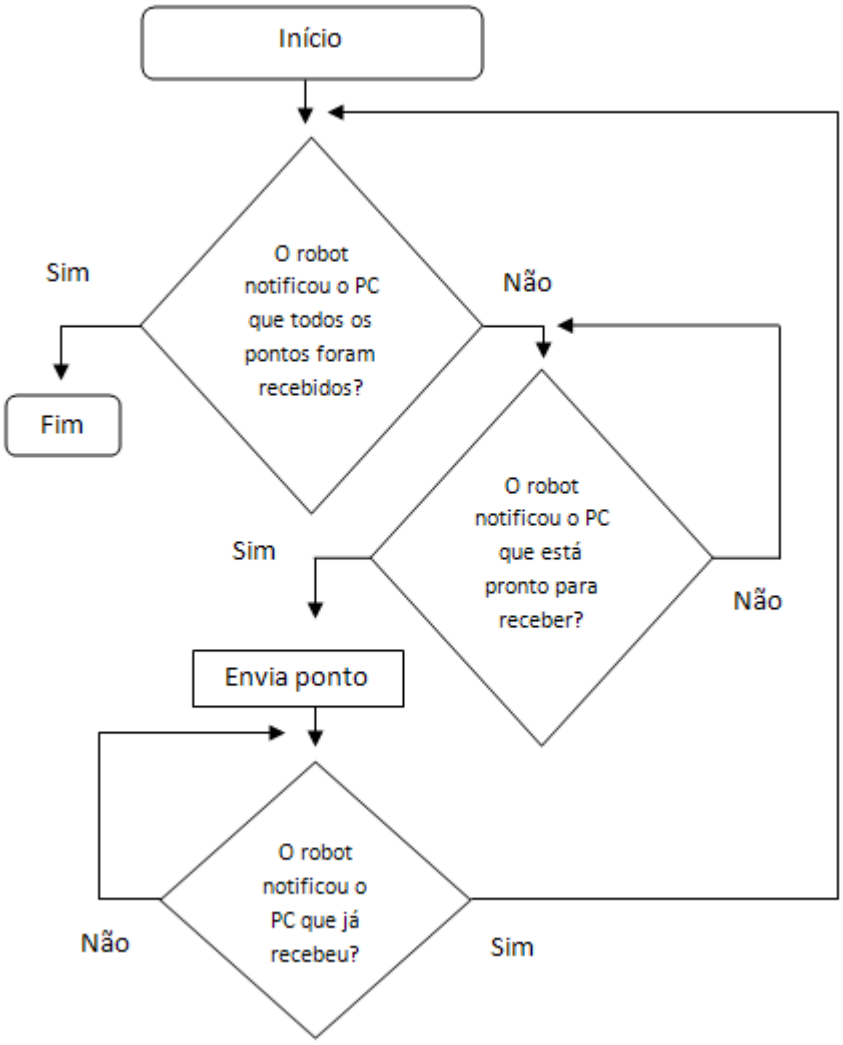


Figura 4.13 - Fluxograma do protocolo de comunicação

O software do fluxograma anterior é apresentado na figura 4.14.

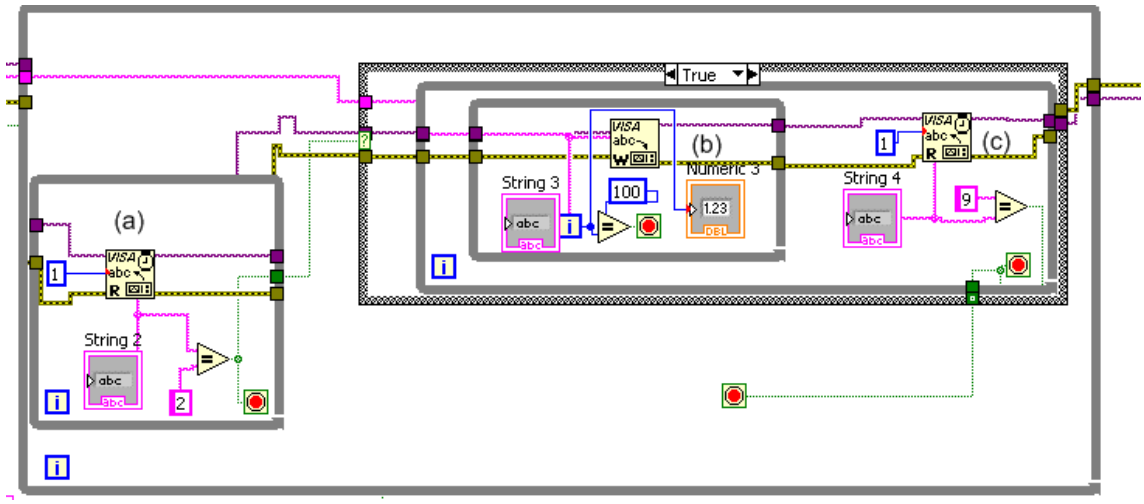


Figura 4.14 - Código do protocolo de envio dos pontos dominantes; (a) – Bloco de leitura; (b) – Bloco de escrita; (c) - Bloco de leitura

Em 4.14 (a) tem-se um ciclo que aguarda que o robot notifique que está pronto para receber, quando este recebe o carácter de notificação do robot, passa para um novo ciclo de escrita do ponto dominante na porta série, (b). Quando o robot terminar de receber o ponto comunica ao computador o ocorrido, (c).

Como dito anteriormente, por cada ponto são enviadas as duas coordenadas, logo, para esse ponto ser tratado como um conjunto de duas coordenadas por parte do robot decidiu-se colocar caracteres de controlo para determinar qual o inicio, meio e fim de cada coordenada enviada. Para isso utilizou-se uma ferramenta que possibilita o emparelhamento das duas coordenadas com os caracteres de controlo (Figura 4.15). Esta ferramenta possibilita, também, a transformação de qualquer um dos dados numéricos em *string*.

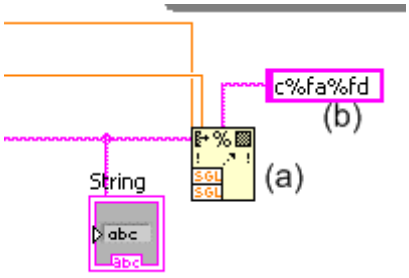


Figura 4.15 - (a) - Bloco de emparelhamento que permite formatar as duas coordenadas em string; (b) - Coordenadas enviadas com caracteres de controlo

Na figura 4.15 (b) pode-se verificar que ambas as coordenadas são representadas por *%f* (numero decimal), estando estas espaçadas de caracteres. O carácter "c" significa o inicio da primeira coordenada, o "a" significa o ponto médio entre as duas coordenadas e o carácter "d" constitui o final das duas coordenadas.

Este pequeno protocolo serve para ajudar o robot a identificar qual as coordenadas correctas a receber. Posteriormente, será apresentada a forma de como o robot recepciona as coordenadas enviadas, serão também fornecidas explicações detalhadas da trama enviada.

4.3 Conclusões

O software Labview utilizado neste projecto revelou-se uma boa ferramenta devido a apresentar já um conjunto de funcionalidades direccionadas para o processamento de imagem o que facilitou toda a tarefa de implementação.

Na comunicação com o robot optou-se pela utilização do protocolo série pela sua simplicidade e facilidade de implementação. A utilização da porta série, neste projecto, revelou-se eficaz na interacção de sistemas complexos como os utilizados.

Devido à comunicação ser feita em apenas um sentido (leitura ou escrita) de cada vez, obrigou à utilização de um protocolo de comunicação, personalizado, entre o computador e o robot.

Capítulo 5

5 Sistema robótico

Os sistemas robóticos são largamente utilizados na indústria hoje em dia. Com a sua utilização é possível a realização de várias tarefas que exigem muitos cuidados e restrições ao serem realizadas pelo homem.

Devido a grandes limitações orçamentais impostas por muitas empresas, cada vez mais este tipo de equipamentos é utilizado. Apesar de existir um investimento inicial, este é recuperado a curto prazo, pois para além de trabalhar mais rapidamente e eficazmente do que o homem, pode realizar trabalho contínuo sem cessar. Com a ajuda dos sistemas robóticos uma entidade industrial pode competir, em vantagem, com outra que não os utilize, pois com estes equipamentos conseguem fornecer mais rápido e produzir melhores produtos para satisfazer o consumidor.

Neste capítulo será apresentado o sistema robótico do projecto, assim como o software implementado para a realização da tarefa de halogenação.

5.1 Aspectos teóricos

Neste projecto foi utilizado o robot IRB 140 da ABB para a realização da tarefa de halogenação. Foi utilizado, também, o controlador IRC 5 e uma consola Flexpendant.

Em traços gerais, a consola Flexpedant serve de interface entre o robot e o utilizador/programador. Através de comandos que o utilizador/programador envia, da consola para o controlador IRC 5 e este de seguida para o robot, é possível controlar o IRB 140 (Figura 5.1).

Figura 5.1 - Imagem do interface Homem/Robot

5.2 Modos de programação

Cada vez mais os robots necessitam de sólidas bases de programação específica para se tornarem funcionais no mundo industrial. Com a ajuda de controladores sofisticados com sistemas operativos próprios, é possível tornar os robots cada vez mais flexíveis.

As programações robóticas são feitas de duas formas:

1. Programação *On-line*;
2. Programação *Off-line*;

5.2.1 Programação *On-line*

A programação *On-line* consiste na interacção directa com o robot, cada vez que o código do programa é testado, o robot executa-o fisicamente, seja programando-o através da Flexpendant ou através de editores de código como o *RobotStudio On-line* (Figura 5.2). No caso do *RobotStudio On-line* o código é editado no software e transferido para o robot, estando disponível na Flexpendant para ser executado. Na Flexpendant, o código é editado na própria e transferido para o robot. Em ambos os casos o uso da Flexpendant é essencial.

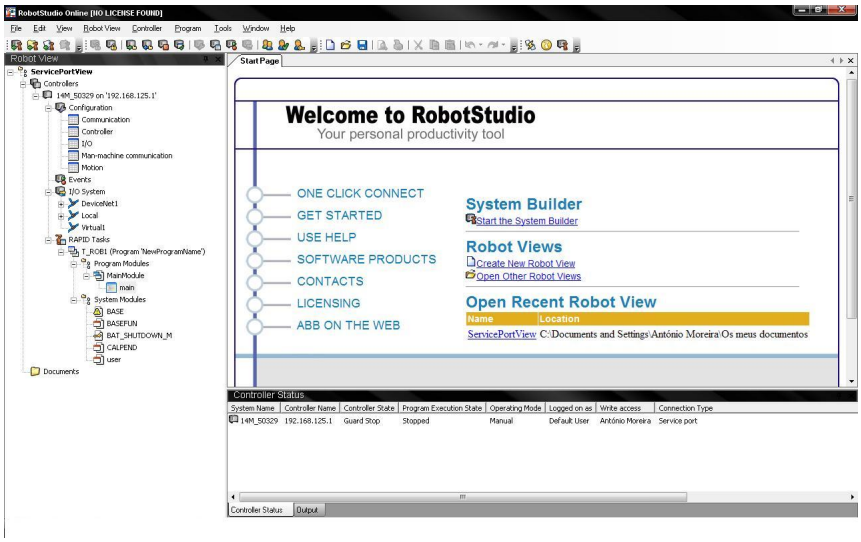


Figura 5.2 - Software RobotStudio On-line

Com a utilização do software *RobotStudio On-line* torna-se mais fácil e rápido editar código, pois existem diversas ferramentas que ajudam o programador a trabalhar mais eficientemente (Figura 5.3).

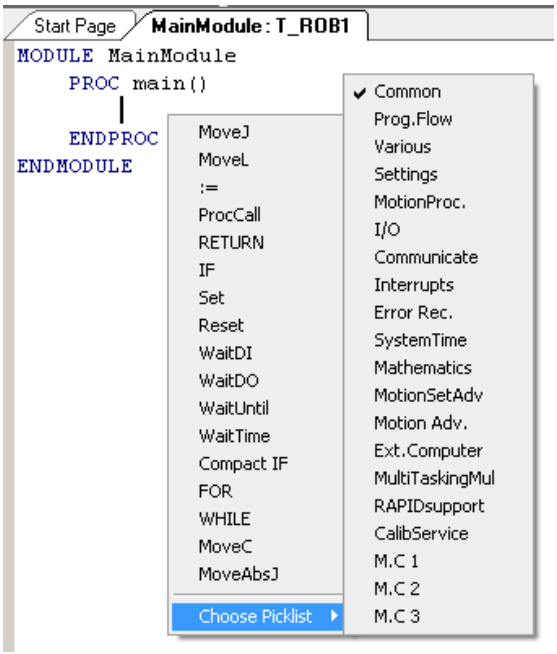


Figura 5.3 - Lista de instruções

Na figura 5.3 é possível verificar que todas as instruções estão rapidamente à disposição do programador, fazendo com que este poupe imenso tempo na edição do código.

No entanto, a programação *On-line* apresenta algumas desvantagens: lentidão, necessidade de equipamento disponível (Robot e controlador) e principalmente o facto de o robot estar parado, sem executar, à espera que o programador insira o código correcto para a sua utilização [18].

5.2.2 Programação *Off-line*

A programação *off-line* permite a utilização de ambientes virtuais para simular a programação do robot, ou seja, com a utilização deste tipo de software é possível o teste virtual do robot. Além disso, nem sequer é necessário estar junto ao robot, pois estes ambientes virtuais permitem uma simulação cada vez mais fiel à realidade.

Um dos sistemas mais utilizados para programação *off-line* é o *RobotStudio 5.12.03*, através deste software é possível a simulação em ambiente virtual (Figura 5.4).

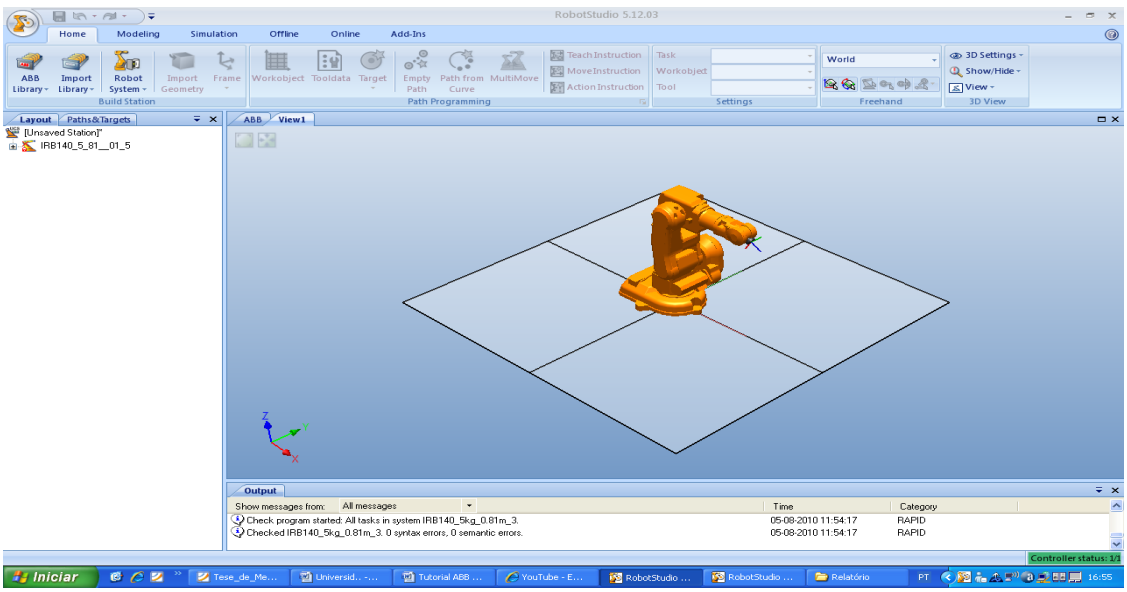


Figura 5.4 - Software RobotStudio 5.12.03

A figura 5.4 mostra o robot IRB 140 na sua posição inicial. Como demonstração vai-se programar, virtualmente, o robot para este se deslocar para um ponto no seu espaço de trabalho.

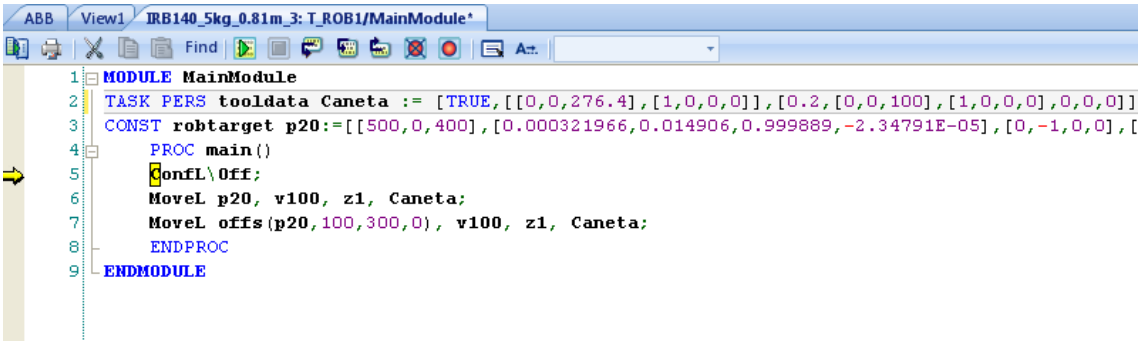


Figura 5.5 - Código introduzido no robot virtual

Na figura 5.5 verifica-se o codigo utilizado no robot virtual. Na linha 2 da imagem declara-se e define-se a ferramenta a ser utilizada, na linha 3 define-se as coordenadas (x, y, z) do ponto *p20*, sendo estas (500,0,400) em mm. Entrando na *main()*, faz-se um movimento linear, linha 6, para o ponto *p20*, a uma velocidade de 100 mm/s (*v100*), numa zona de trabalho (*z1*), com uma ferramenta (*caneta*). Movendo o robot para o ponto (500,0,400) obtém-se a figura 5.4, anteriormente apresentada.

Na linha 7 é realizado, novamente, um movimento linear, mas com a ajuda da função *offset*, que insere deslocamentos em (x, y, z) em relação ao ponto introduzido. Ou seja, no exemplo da figura 5.5 será deslocado 100 mm em x e 300 mm em y, apartir do ponto 20, obténdo-se a figura 5.6.

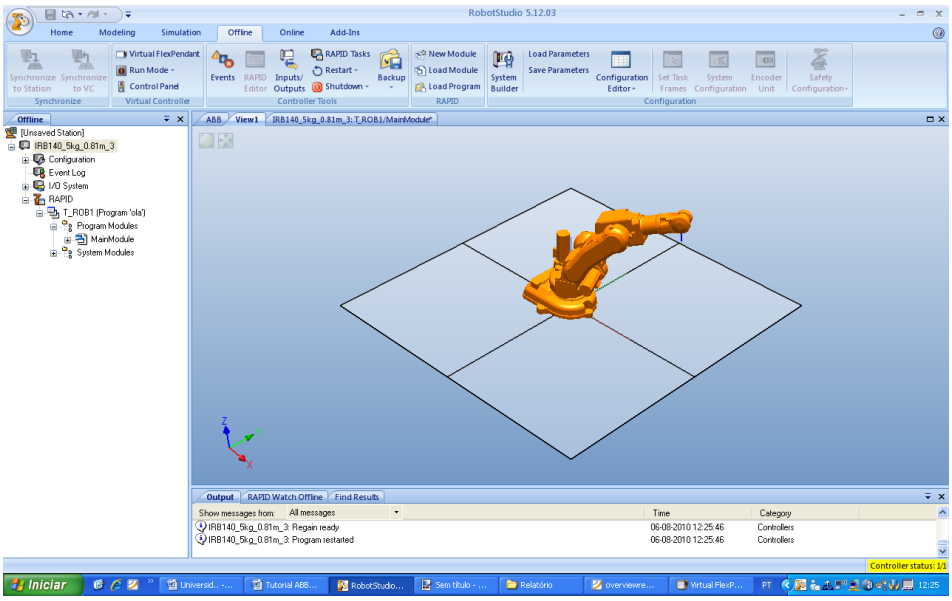


Figura 5.6 - Resultado do código introduzido no robot

Além do editor de código da figura 5.5, existe uma Flexpendant virtual com todas as funcionalidades das consolas reais (Figura 5.7).

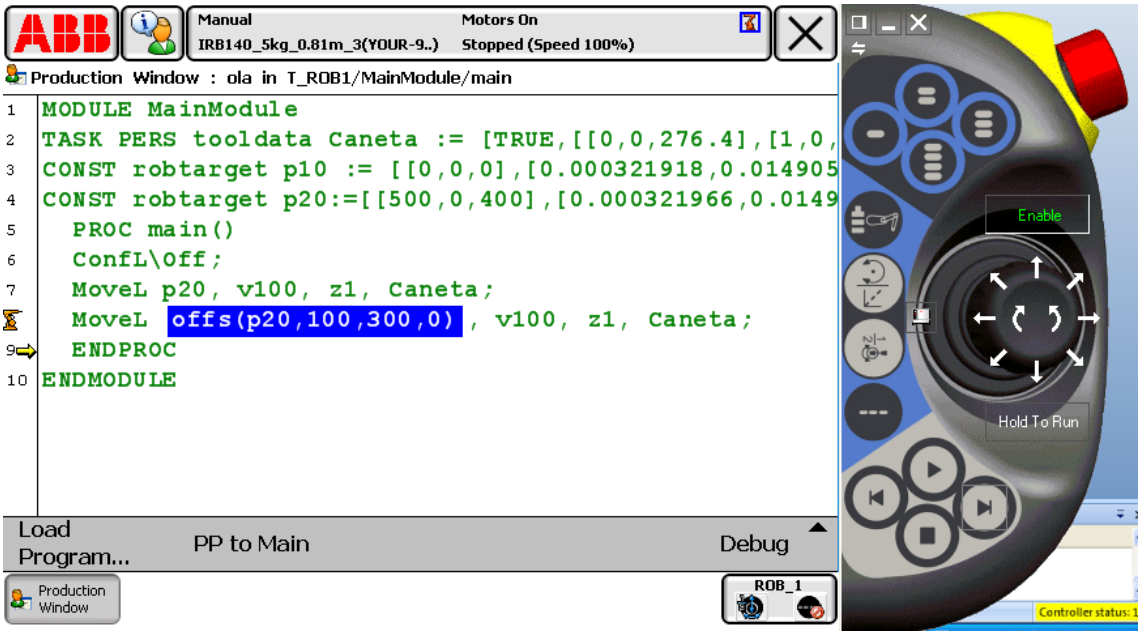


Figura 5.7 - Flexpendant virtual

5.3 Software do manipulador

Nesta secção será apresentado o algoritmo utilizado para a realização do processo de halogenação e o código implementado no robot. A linguagem utilizada para programar o robot foi o RAPID que é uma linguagem de alto nível para uso exclusivo nos robots da ABB.

5.3.1 Algoritmo

O algoritmo explicado nesta secção contém duas etapas distintas. A primeira refere-se à explicação de como são recepcionados os pontos dominantes da sola e seu respectivo tratamento de software. Enquanto a segunda diz respeito ao deslocamento do robot pelos pontos recepcionados.

Como foi dito no esclarecimento do protocolo de envio em 4.2.2, as coordenadas de cada ponto dominante são enviadas numa *string*. A *string* enviada contém quer ambas as coordenadas x e y de cada ponto, como os caracteres de controlo que lhe dão inicio, meio e fim. Recorde-se que carácter "c" significa o inicio da primeira coordenada, o "a" significa o ponto médio entre as duas coordenadas e o carácter "d" constitui o final das duas coordenadas (Figura 5.8).

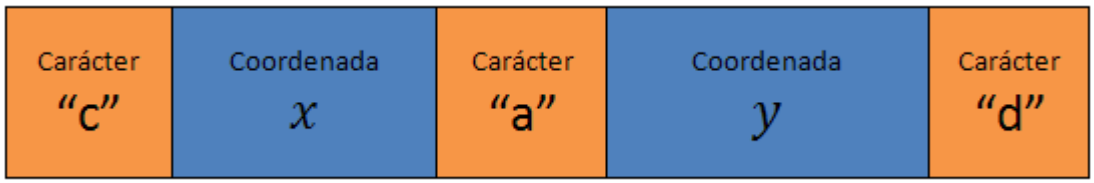


Figura 5.8 - Trama a enviar para cada ponto

O fluxograma seguinte exemplifica o modo como o robot recepciona as tramas enviadas pelo computador (Figura 5.9).

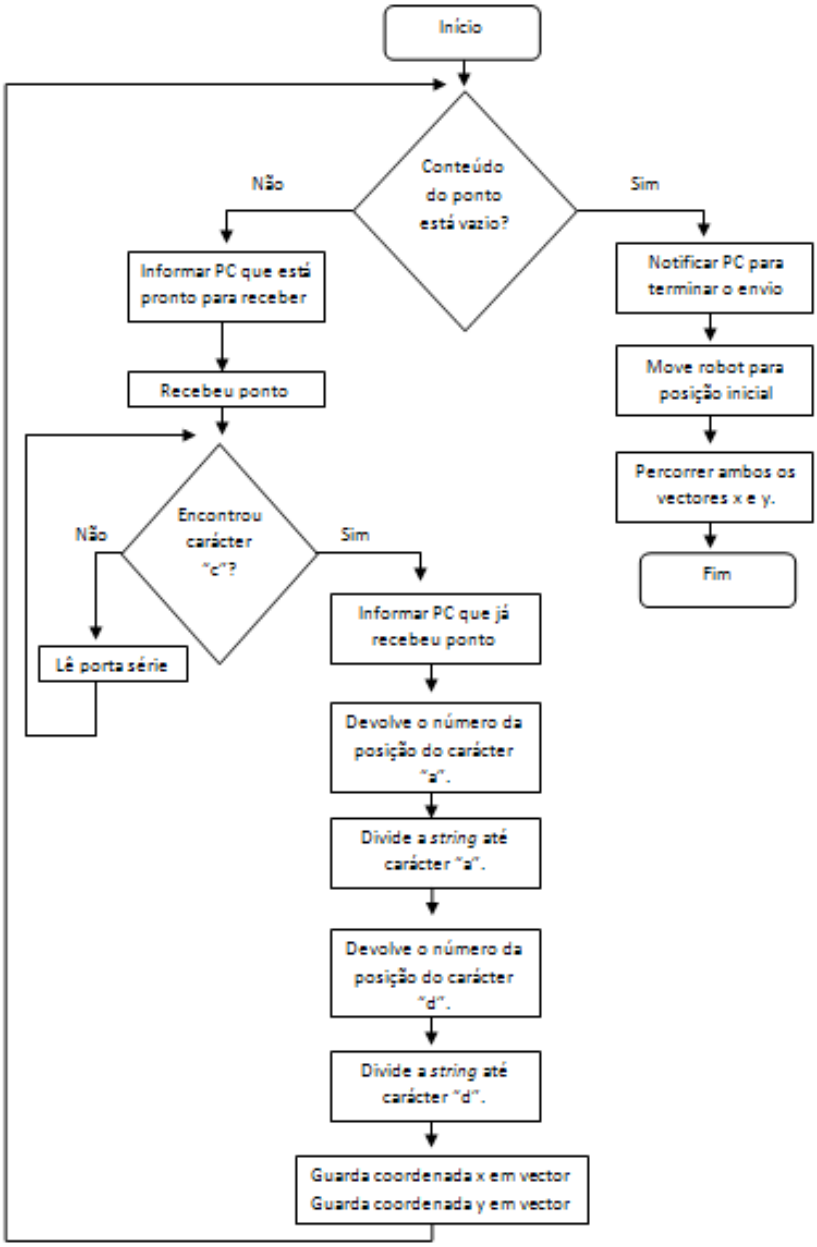

Início
Conteúdo do ponto está vazio?
Não
Sim
Informar PC que está pronto para receber
Recebeu ponto
Encontrou carácter "c"?
Não
Sim
Lê porta série
Informar PC que já recebeu ponto
Devolve o número da posição do carácter "a".
Divide a string até carácter "a".
Devolve o número da posição do carácter "d".
Divide a string até carácter "d".
Guarda coordenada x em vector
Guarda coordenada y em vector
Notificar PC para terminar o envio
Move robot para posição inicial
Percorrer ambos os vectores x e y.
Fim

Figura 5.9 - Funcionamento do software do robot

O robot inicialmente informa o PC que está pronto para receber um ponto. Ao ser recebido o software deve procurar o carácter "c", pois só aí a *string* é recebida correctamente. Após a sua a recepção, deve ser procurado o carácter "a", pois este serve como linha divisória entre a primeira e segunda coordenada. Ao ser encontrado, a *string* é partida em duas, tendo a primeira início no carácter "c" até ao carácter "a", excluindo ambos, obtendo a coordenada x. A segunda inicia-se no carácter "a" e termina no "d", excluindo novamente ambos os caracteres, obtendo a coordenada y. Por último, basta

transformar ambas as coordenadas em números, visto que estas ainda são *strings*, e comunicar ao PC que já recebeu o primeiro ponto e que este pode enviar o próximo. Este processo repete-se para todos os pontos dominantes.

Após a recepção e tratamento de cada ponto dominante estes são guardados num vector, respectivo, para cada orientação. Quando o robot detectar a recepção de *strings* vazias, enviadas pelo computador, significa que todos os pontos foram enviados. Neste caso, o robot notifica o computador para cessar comunicações. De seguida, o robot é movido para a sua posição inicial e percorridos ambos os vectores x e y. Realizando, assim, a simulação da tarefa de halogenação.

5.3.2 Código do manipulador

```
Proc main()
        Confl\off
        Xdata:=1;
        While Xdata<>"" do

                Open "com1", porta\write;
                Write porta, string1;                    } 1
                Close porta;

                While found3<>"c" do
                        Open "com1", porta\read;         } 2
                        Text1:=readstr(porta);
                        Close porta;
                End while

                Open "com1", porta\write;
                Write porta, string3;                    } 3
                Close porta;

                Found:=strmatch(text1,1,"a");
                Xdata:=strpart(text1,1,found);           } 4
                Found1:=strmatch(text1,1,"d");
                Ydata:=strpart(text1,found,found1);
```

```
        If Xdata:="" then
        Else
            Flag:=strtoval(xdata,x{i});
            Flag:=strtoval(ydata,y{i});
        endif

        Found3=:" ";
    j:=j+1;
    End while

    Open "com1", porta\write;
    Write porta, string2;
    Close porta;
    j:=j-1;
    Movel p10,v50,z1,tool;

    For i from 1 to j do

        Movel offs(p10,x{i},y{i},-80),v50,z1,tool;

    End for

End proc
```

5

6

1. Nesta parte do código é onde se notifica o PC que o robot está pronto a receber.
2. Aqui é recebido o ponto dominante, enquanto não for encontrado o carácter "c" este abre a porta série em modo de leitura e armazena o ponto em *text1*. Quando encontrar o carácter sai do ciclo.
3. Esta parte é exactamente igual à explicada em 1, e serve para notificar o PC que o ponto anterior foi recebido.
4. Na primeira linha da parte 4 é percorrida a *string* do ponto recebido. É devolvido o número da posição do carácter "a", sendo esse valor armazenado em *found*. Na segunda linha a *string* que contém o ponto é dividida desde o seu carácter 1 até ao valor armazenado em *found.* Até ao momento já se tem a coordenada *x* armazenada em *xdata*. As duas linhas de código seguintes fazem exactamente o que foi explicado anteriormente, excepto na verificação da posição do carácter.

Agora, a verificação faz-se até ao carácter “d”, dividindo-se a *string* desde a posição do carácter ”a” até ao “d”, armazenando o valor da coordenada *y* em *ydata*.

5. Aqui as *strings*, de ambas as coordenadas, são transformadas em valores numéricos e armazenadas nos seus respectivos vectores, excepto quando *Xdata* não tem qualquer valor armazenado. Desta forma garante-se que não acontecem erros na execução.

Note-se que as partes 1 à 5 estão dentro de um ciclo, onde é feito o teste da variável armazenada, com a orientação *x* do ponto recepcionado. Caso essa variável esteja vazia, significa que o computador já enviou todos os pontos. Logo, a condição, testada no ciclo, é atingida e executada a parte 6 do código.

6. Quando todos os pontos são recebidos e armazenados, o robot notifica o computador para cessar comunicações. De seguida, o manipulador é movido para a posição inicial, entrando num ciclo que percorrerá os pontos de contorno a partir do seu ponto inicial.

5.4 Conclusões

Com a realização deste capítulo, foi possível concluir que o robot IRB 140 é uma boa solução para o sistema implementado devido à sua flexibilidade, velocidade e dimensões.

No que diz respeito à programação do robot pode ser efectuada de duas formas:

- Programação *On-line*;
- Programação *Off-line*;

Devido a problemas de comunicação não foi possível passar o código do *Robotstudio* para o robot, pois a versão utilizada era um *trial*. Tornando-se inviável a utilização do modo *off-line*, toda a programação realizada foi *On-line*, sendo utilizada a Flexpendant para o efeito.

Foi, também, necessária a implementação de um algoritmo e consequente código para a recepção dos pontos dominantes por parte do robot.

Capítulo 6

6 Resultados e Conclusões

Neste capítulo serão apresentados os resultados finais e as conclusões do trabalho realizado.

6.1 Resultados

6.1.1 Interface LabView

Inicialmente pensou-se utilizar um interface, realizado em LabView, que executa as técnicas de processamento de imagem passo a passo consoante a pressão, por parte do utilizador, de botões associados a cada uma delas. Utilizando este tipo de interface é possível, ao utilizador, validar o efeito de cada uma das técnicas implementadas (Figuras 6.1, 6.2, 6.3, 6.4 e 6.5).

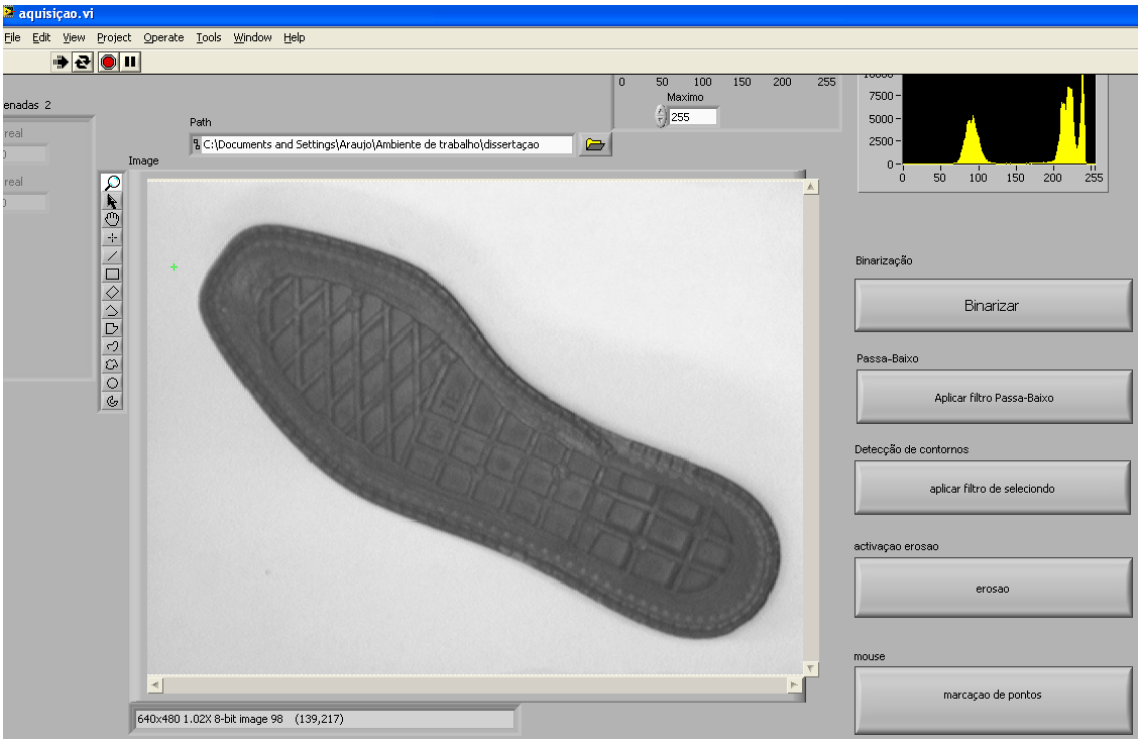


Figura 6.1 - Aquisição da imagem

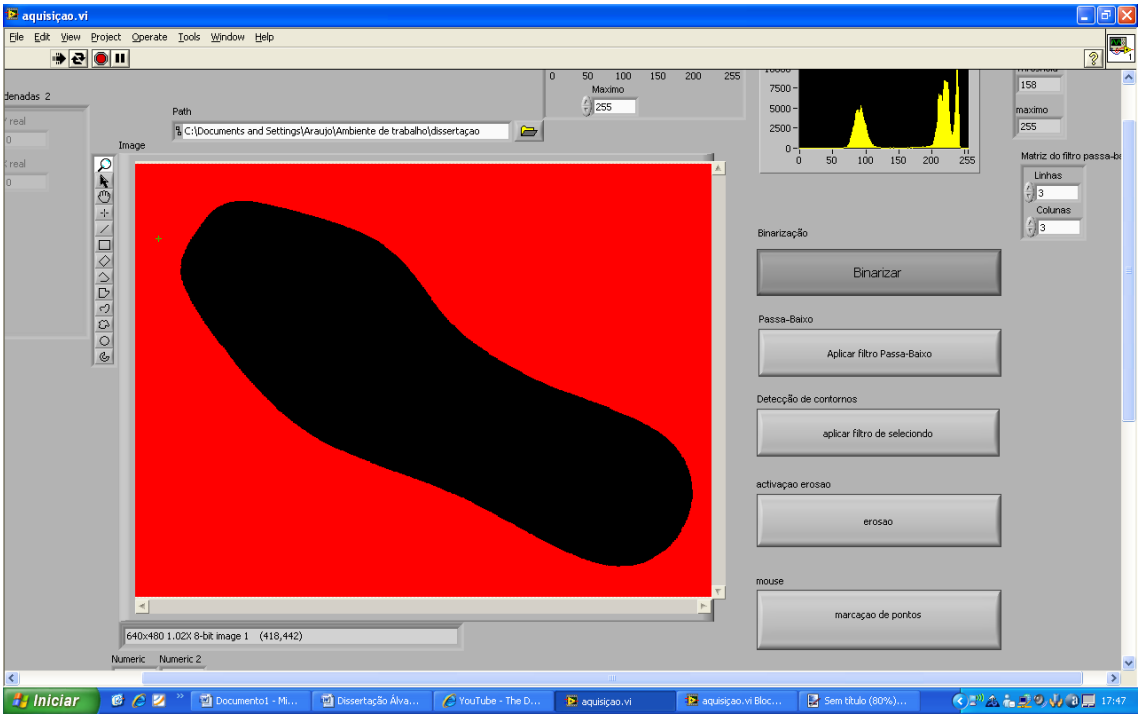


Figura 6.2 - Conversão binária

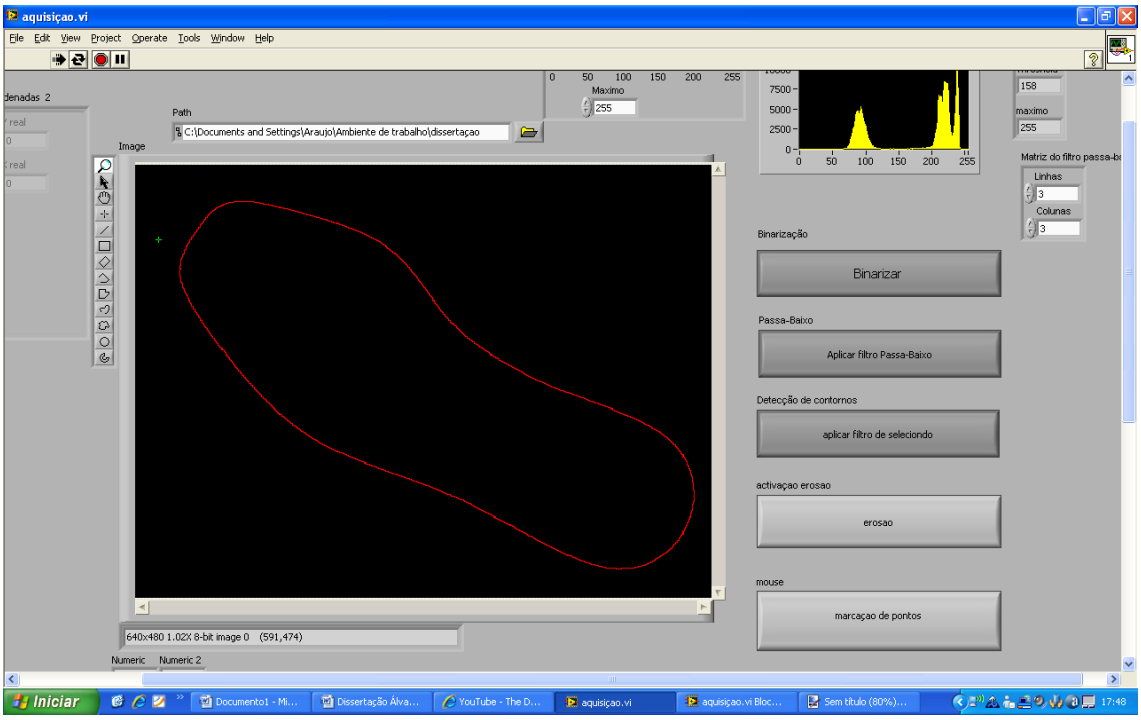


Figura 6.3 -Detecção de contornos

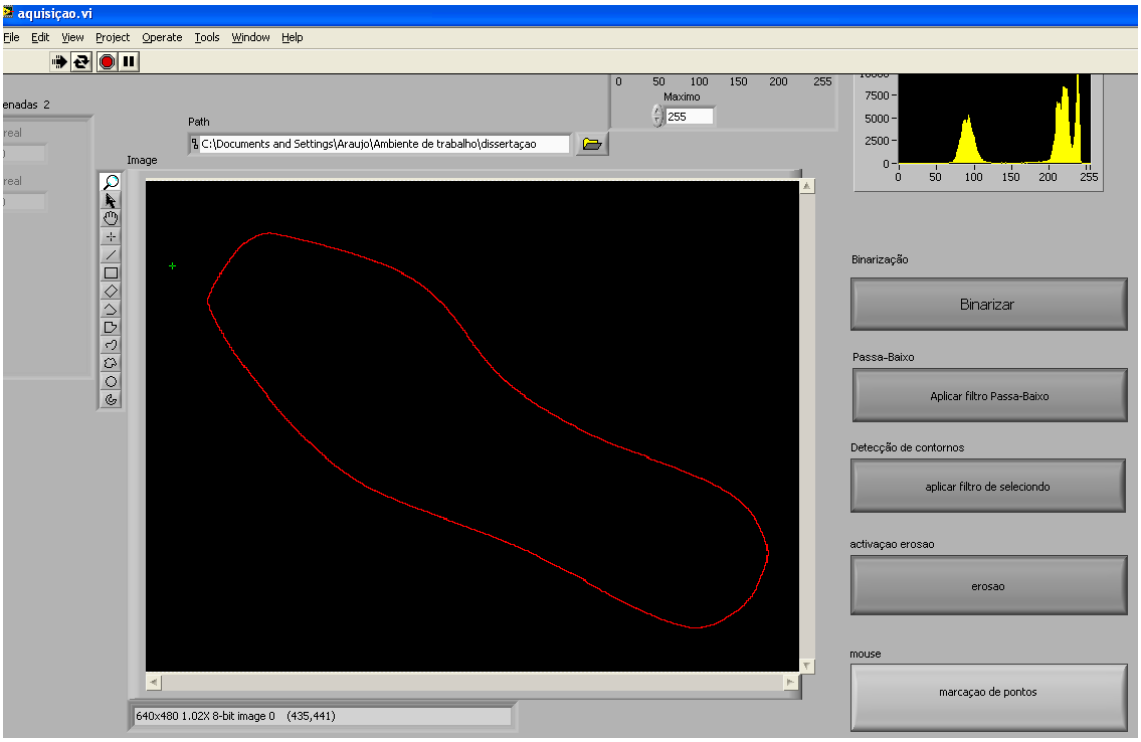


Figura 6.4 - Erosão

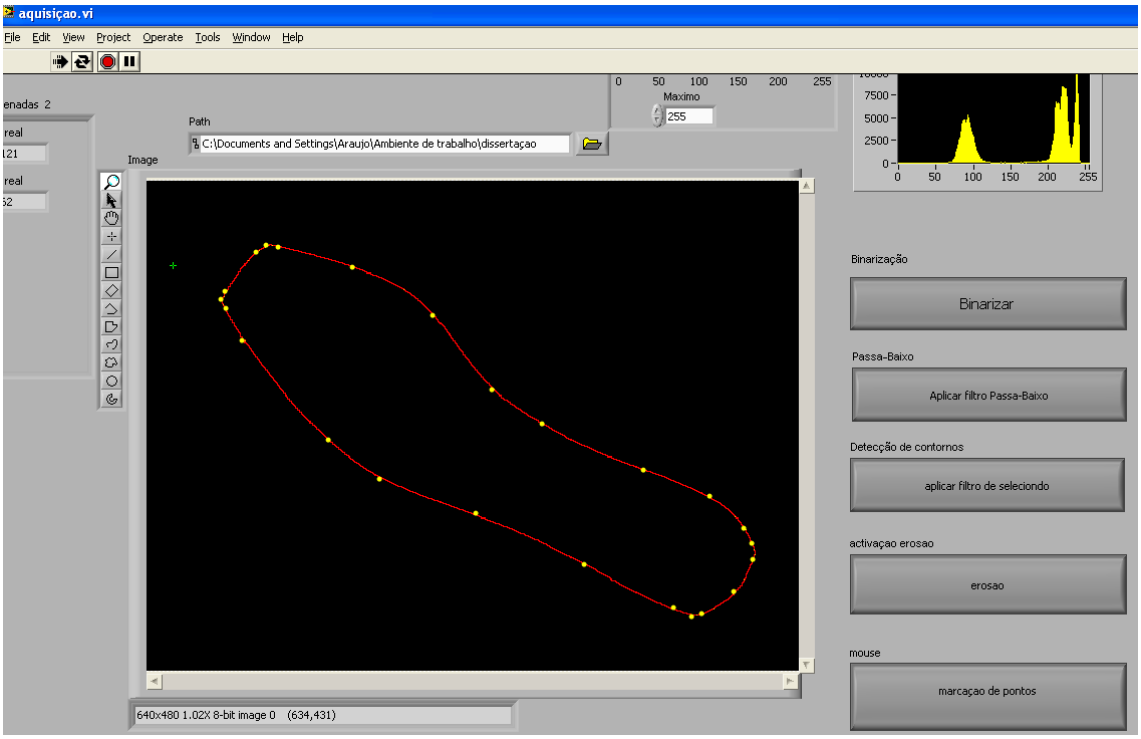


Figura 6.5 - Marcação de pontos dominantes

Como versão final optou-se por desenvolver um interface mais amigável e simplificado. Verificou-se, também, que seria contraproducente a utilização de muitos botões para o utilizador pressionar, resultando em perda de tempo de execução. Uma solução possível foi a utilizada na figura 6.6, onde apenas existem 2 botões. O primeiro serve para marcar os pontos dominantes e o segundo para enviar esses pontos para o robot. As técnicas de processamento de imagem são executadas automaticamente.

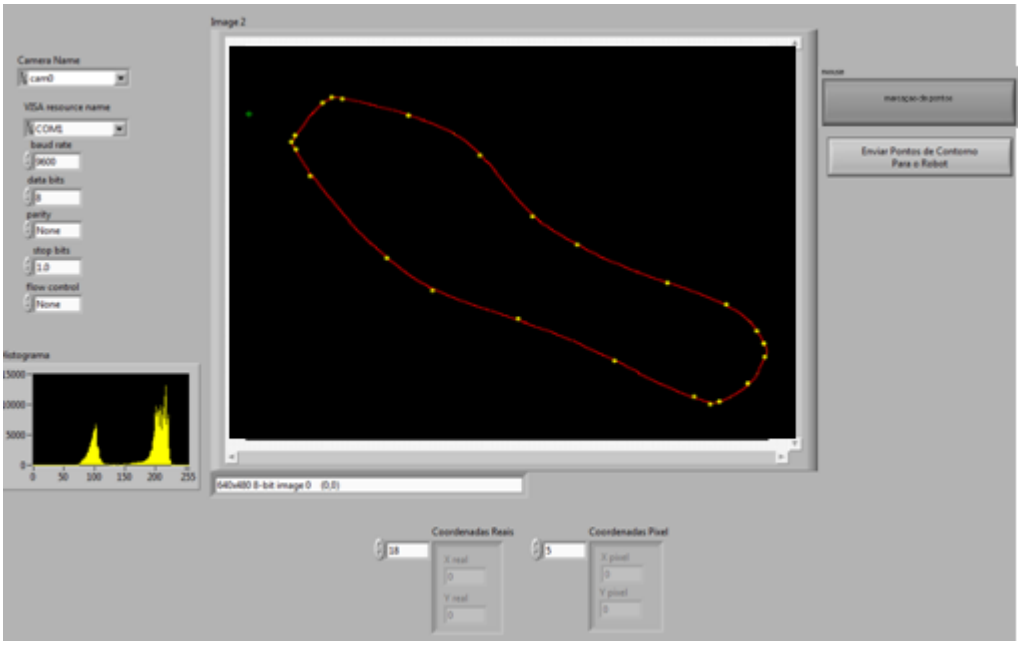


Figura 6.6 - Ambiente utilizado

6.1.2 Sistema robótico

Para executar o código do manipulador é necessário, também, ter disponibilizado na porta série os pontos dominantes. Logo, pode-se concluir que o sistema robótico é dependente da visão por computador.

Aqui podem-se verificar algumas imagens do robot a executar a tarefa, Figuras 6.7, 6.8 e 6.9.

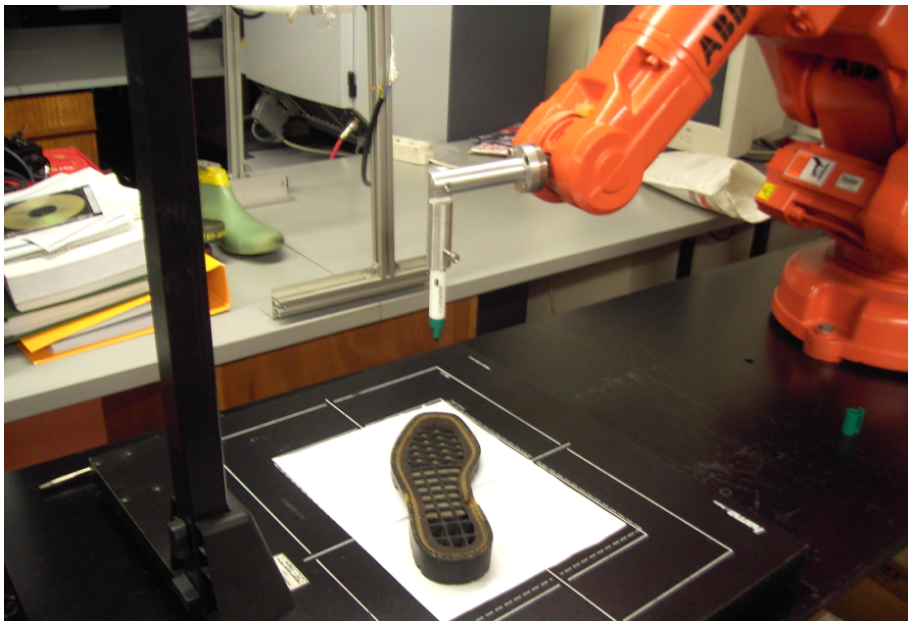


Figura 6.7 - Robot na posição inicial

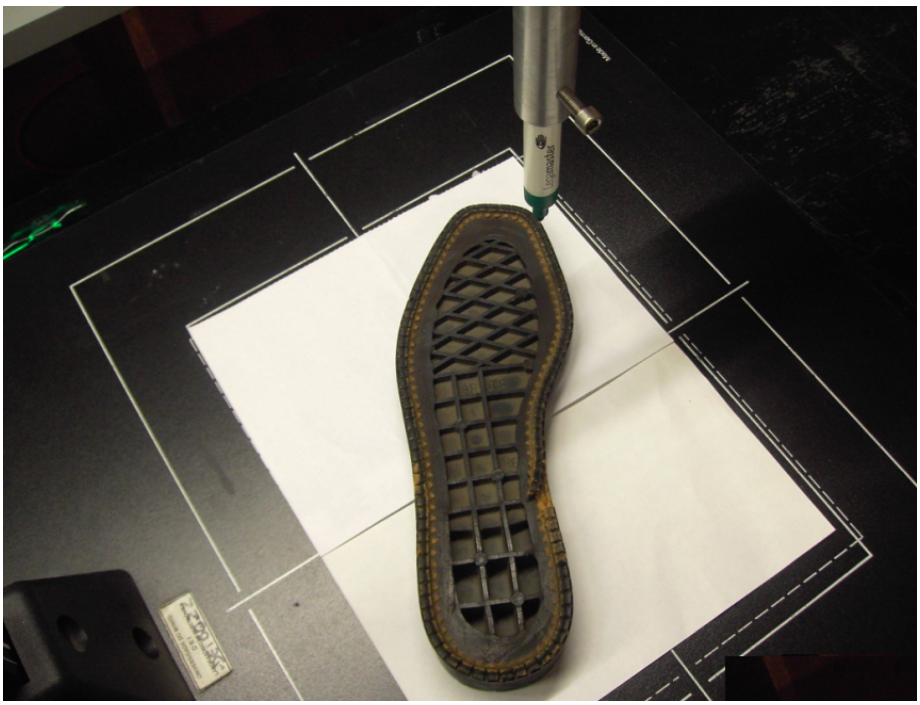


Figura 6.8 - Execução da tarefa

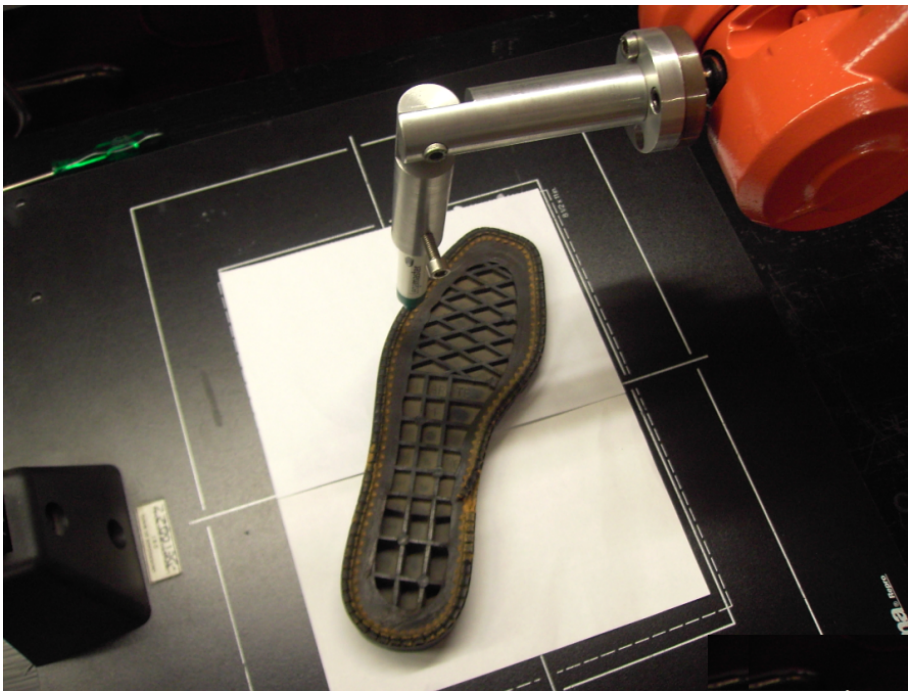


Figura 6.9 - Execução da tarefa

6.2 Conclusões

Cada vez mais são utilizados, na produção de calçado, ferramentas como a robótica e a visão por computador. Com a ajuda destas ferramentas é possível um melhoramento considerável na qualidade do calçado produzido, pois os produtos ficam menos susceptíveis a erro humano.

A plataforma de programação gráfica Labview foi utilizada em conjunto com a visão por computador. Durante a pesquisa realizada na visão por computador notou-se a existência de inúmeros documentos com linguagens de programação demasiadamente complicadas e com inúmeras linhas de código. A utilização do Labview facilitou a implementação do sistema de visão devido as funcionalidades que dispõe neste âmbito. Além disso, mostrou ser um software que produz ambientes gráficos bastante esclarecedores, ajudando as pessoas pouco qualificadas na sua utilização. As ferramentas da visão por computador revelaram-se muito competentes, pois, possibilitaram uma melhor visualização da imagem (filtros), redução dos dados a serem processados (detecção de contornos) e transformação dos pontos dominantes em

coordenadas do mundo real. Apesar de existirem alguns algoritmos para a marcação dos pontos dominantes como em 3.3.6.1 e 3.3.6.2, decidiu-se a utilização de um método diferente dos anteriormente mencionados.

O robot IRB140 revelou-se uma boa opção para a execução desta tarefa. Devido às suas dimensões e velocidade de execução tornam este robot ideal para a fazer parte integrante do processo produtivo de uma empresa.

Um dos maiores desafios da realização desta dissertação é, sem dúvida, a possibilidade de contribuir para a saúde e o bem-estar dos trabalhadores fabris. O uso indevido ou nulo de mascaras e luvas protectoras na utilização de materiais tóxicos na indústria ainda é grande hoje em dia. Uma das principais conclusões que se pode realçar é o facto de a utilização de braços robóticos poderem substituir os trabalhadores nestas tarefas industriais.

Finalmente pode-se afirmar que todos os objectivos propostos em 1.2 foram cumpridos, sem excepção.

6.2.1 Trabalho futuro

Como trabalho futuro, seria muito interessante possibilitar ao utilizador deste sistema, uma forma de consulta de históricos de execução da tarefa. Ou seja, criar-se uma base de dados com a quantidade de solas que foram halogenadas, qual o seu número e qual a data e hora da execução.

Referências

[1] Vilaca, J.L. Fonseca, J., "A New Software Application for Footwear Industry", Univ. of Minho, Guimarães, 2007 – Intelligent Signal Processing, 2007. WISP 2007. IEEE International Symposium on ISBN: 978-1-4244-0830-6. p. 1-6.

[2] João Sepulveda, Carlos Couto, João Monteiro, "Developing an Automated System for Shoe Sole Halogenation", Guimarães, 2007.

[3] Welt Sewing Machine. Goodyear. Disponível em http://www.tesmc.com/detail.aspx?pid=2. Consultado a 27 de Fevereiro de 2010.

[4] Thermoadesive Attaching Machine. Goodyear. Disponível em http://www.tesmc.com/detail.aspx?pid=41. Consultado em 28 de Fevereiro de 2010.

[5] Inseam Trimming Machine. Goodyear. Disponível em http://www.tesmc.com/detail.aspx?pid=6. Consultado em 28 de Fevereiro de 2010.

[6] Lirel. Disponível em http://www.lirel.pt/. Consultado em 30 de Março de 2010.

[7] Lirel. Disponível em http://www.lirel.pt/html/calcado-manipulacao_robotizacao.htm. Consultado em 5 de Abril de 2010.

[8] Manuel J. Ferreira, Aparício Fernandes, J.A. Campos Neves, João L. Monteiro, "Automatic Visual Inspection on Shoe and Textile Industries", IASTED – CGIM'98, 1998, Halifax - Canada.

[9] Zhongxu Hu, Chris Marshall, Robert Bicker, Paul Taylor, "Automatic surface roughing with 3D machine vision and cooperative robot control" Robotics and Autonomous Systems Volume 55, Issue 7 (July 2007) Pages: 552-560 Year of Publication: 2007 ISSN: 0921-8890.

[10] Luiz Sugimoto, Jornal da Unicamp, "Sistema detecta mudanças de cenário em plataformas de veículos espaciais", Campinas, 15 a 31 de Dezembro de 2008 – ANO XXIII – Nº 419. Disponivel em http://www.unicamp.br/unicamp/unicamp_hoje/ju/dezembro2008/ju419_pag05.php#. Consultado a 22 de Fevereiro de 2010.

[11] GALO, Mauricio; TOMMASELLI, A. M. G.; HASEGAWA, J. K. "Avaliação geométrica de câmaras de vídeo para uso em fotogrametria digital". In: XIX Congresso Brasileiro de Cartografia. Recife – PE, 1999.

[12] Daniel Carneiro da Silva, "Rectificação de imagens digitais de fachadas com pré-correcção das distorções das lentes", Bol. Ciênc. Geod., sec. Comunicações, Curitiba, v. 12, no1, p.123-138, jan-jun, 2006.

[13] Adobe Photoshop CS4, Distorção de lente. Disponível em http://help.adobe.com/pt_BR/Photoshop/11.0/WS04BE5944-BF2A-41c9-84D8-3D447FF965CEa.html. Consultado em 18 de Maio de 2010.

[14] The Mathworks , "Image Processing Toolbox", Agosto de 2004.

[15] Rafael C. Gonzalez, Richard E. Woods, Steven L. Eddins, "Digital Image Processing Using Matlab", 2004.

[16] Martin A. Fischler, Oscar Firschein, "Readings in Computer Vision Issues, Problems, Principals and Paradigms".

[17] Román Katz and Claudio Delrieux, "Boundary Extraction Through Gradient-Based Evolutionary Algorithm", Departamento de Ingenieria Electrica y de Computadoras Universidad Nacional del Sur, Bahia Blanca, ARGENTINA.

[18] Vitor Santos Botazzi, "Framework para programação *off-line* de robots", Departamento de electrónica industrial, Escola de Engenharia, Universidade do Minho, Braga 2006.

[19] ABB Flexible Automation, "System Data Types and Routines On-line", RAPID Reference Manual.

[20] IRB 140 Industrial Robot ABB. ABB datasheet 2010. Disponível em http://www05.abb.com/global/scot/scot241.nsf/veritydisplay/b7d0e4afc2c18435c125772e005530ee/$File/IRB%20140%20PR10031EN%20R10.pdf.

[21] IRC 5 Industrial Robot controller. ABB datasheet 2010. Disponível em http://www05.abb.com/global/scot/scot241.nsf/veritydisplay/da1ec30975d92fedc1257706006d1080/$File/IRC5%20datasheet%20PR10258%20EN_R11.pdf.

[22] IRC5 com Flexpendant. Manual do utilizador. ABB, 2004

ANEXOS

Anexo I – Local de trabalho e materiais utilizados

Figura 7.1 - Controlador IRC 5 e Flexpendant

Figura 7.2 - Mesa de trabalho

Figura 7.3 - Robot IRB140

Figura 7.4 - Suporte de iluminação, câmara e sola de sapato

Anexo II – Funções utilizadas

Instructions **Open**

Open **Opens a file or serial channel**

Open is used to open a file or serial channel for reading or writing.

Example

```
VAR iodev logfile;
.
Open "flp1:LOGDIR/" \File:= "LOGFILE1.DOC",logfile;
```

The file *LOGFILE1.DOC* in unit *flp1:* (diskette), directory *LOGDIR*, is opened for writing. The reference name *logfile* is used later in the program when writing to the file.

Arguments

Open Object [\File] IODevice [\Read] | [\Write] | [\Append] | [\Bin]

Figura 7.5 - Função "Open" [19]

Instructions **Write**

Write **Writes to a character-based file or serial channel**

Write is used to write to a character-based file or serial channel. The value of certain data can be written as well as text.

Examples

```
Write logfile, "Execution started";
```

The text *Execution started* is written to the file with reference name *logfile*.

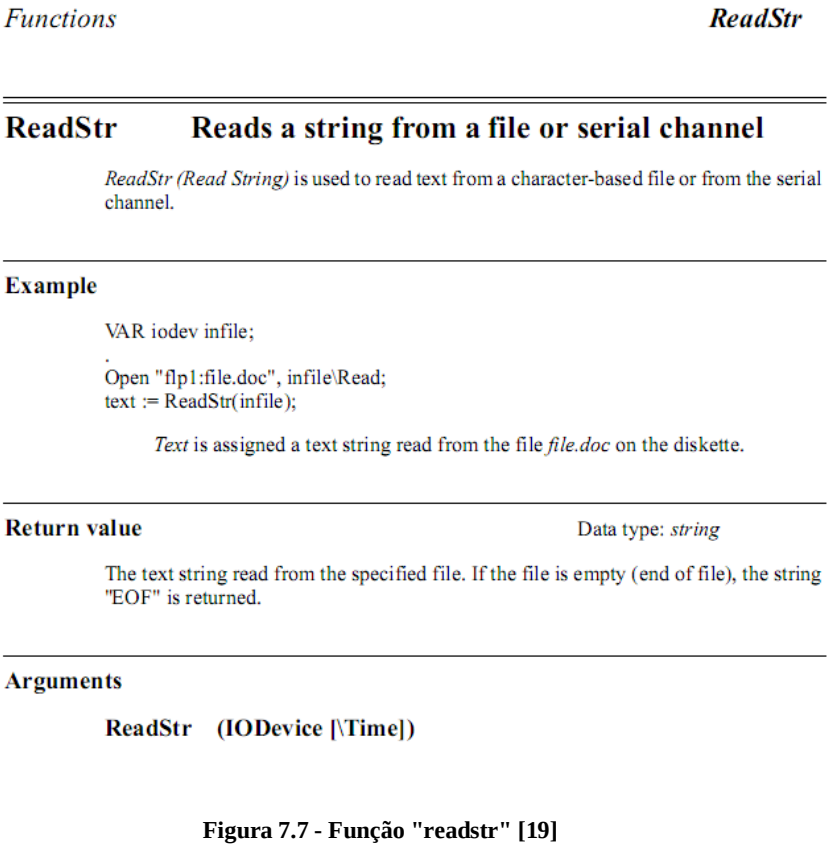
```
Write logfile, "No of produced parts="\Num:=reg1;
```

The text *No of produced parts=5*, for example, is written to the file with the reference name *logfile* (assuming that the contents of *reg1* is 5).

Arguments

Write IODevice String [\Num] | [\Bool] | [\Pos] | [\Orient] [\NoNewLine]

Figura 7.6 - Função "write" [19]



Functions ***ReadStr***

ReadStr **Reads a string from a file or serial channel**

ReadStr (Read String) is used to read text from a character-based file or from the serial channel.

Example

```
VAR iodev infile;
.
Open "flp1:file.doc", infile\Read;
text := ReadStr(infile);
```

Text is assigned a text string read from the file *file.doc* on the diskette.

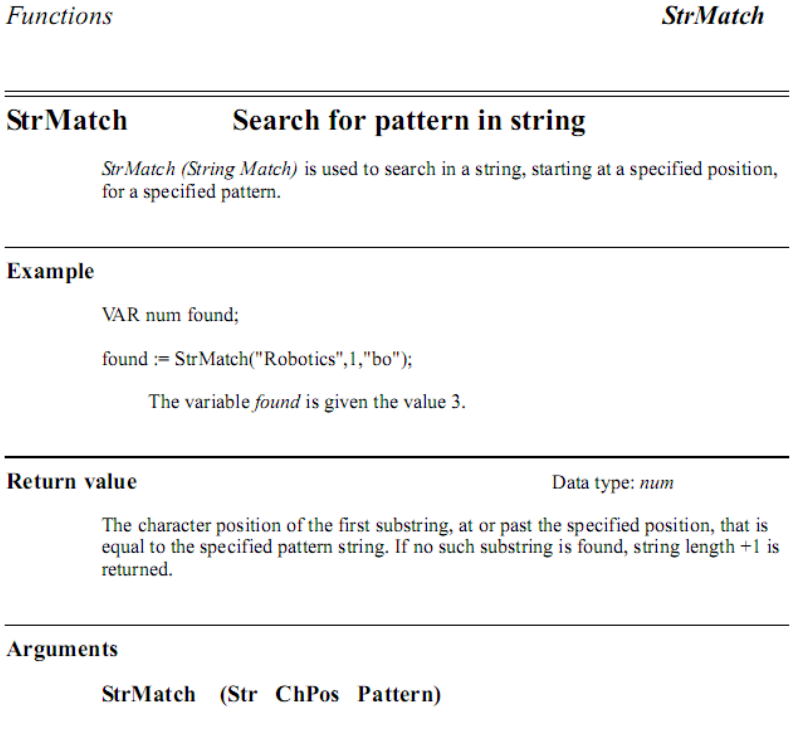
Return value Data type: *string*

The text string read from the specified file. If the file is empty (end of file), the string "EOF" is returned.

Arguments

ReadStr (IODevice [\Time])

Figura 7.7 - Função "readstr" [19]



Functions ***StrMatch***

StrMatch **Search for pattern in string**

StrMatch (String Match) is used to search in a string, starting at a specified position, for a specified pattern.

Example

```
VAR num found;

found := StrMatch("Robotics",1,"bo");
```

The variable *found* is given the value 3.

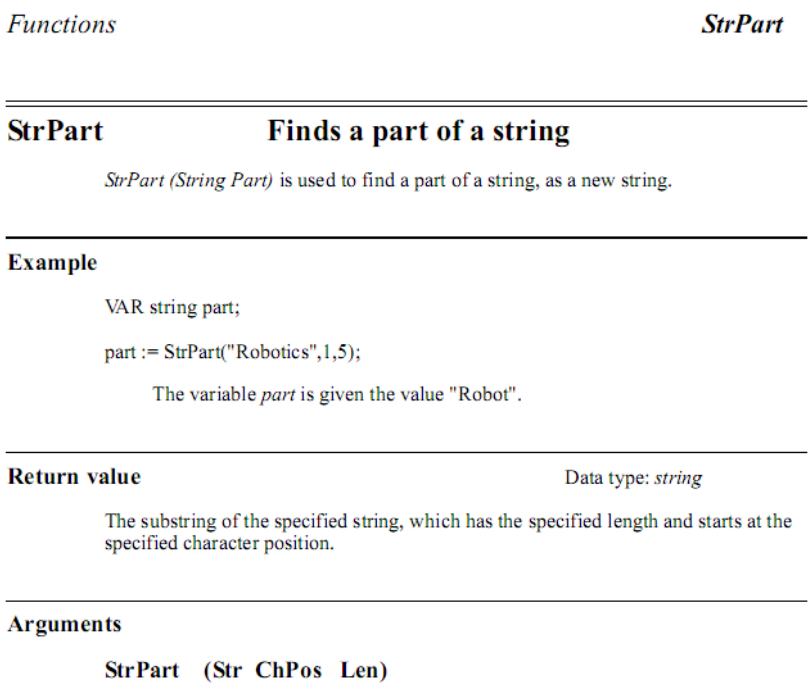
Return value Data type: *num*

The character position of the first substring, at or past the specified position, that is equal to the specified pattern string. If no such substring is found, string length +1 is returned.

Arguments

StrMatch (Str ChPos Pattern)

Figura 7.8 - Função "strmatch" [19]



Functions ***StrPart***

StrPart **Finds a part of a string**

StrPart (String Part) is used to find a part of a string, as a new string.

Example

```
VAR string part;

part := StrPart("Robotics",1,5);
```

The variable *part* is given the value "Robot".

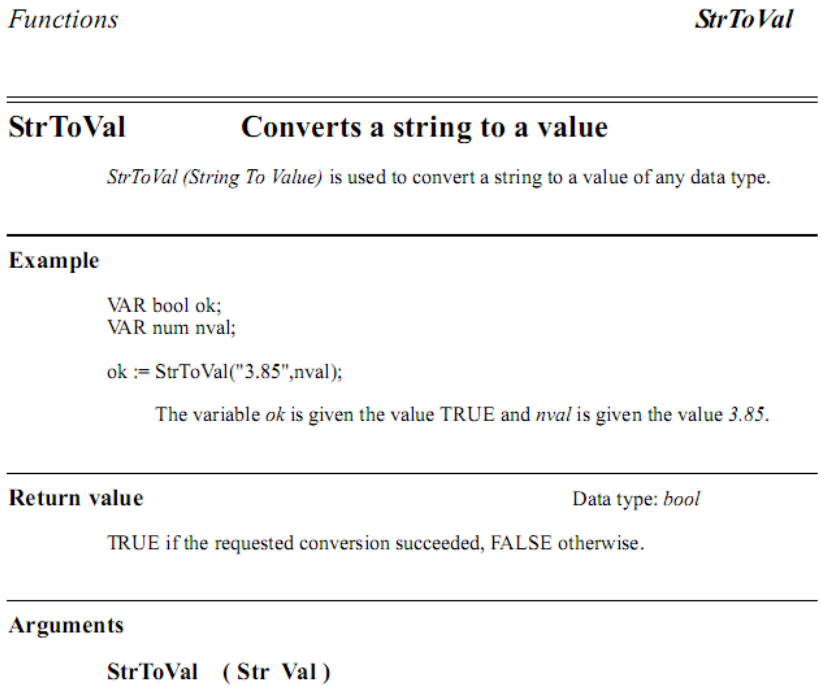
Return value Data type: *string*

The substring of the specified string, which has the specified length and starts at the specified character position.

Arguments

StrPart (Str ChPos Len)

Figura 7.9 - Função "strpart" [19]



Functions ***StrToVal***

StrToVal **Converts a string to a value**

StrToVal (String To Value) is used to convert a string to a value of any data type.

Example

```
VAR bool ok;
VAR num nval;

ok := StrToVal("3.85",nval);
```

The variable *ok* is given the value TRUE and *nval* is given the value *3.85*.

Return value Data type: *bool*

TRUE if the requested conversion succeeded, FALSE otherwise.

Arguments

StrToVal (Str Val)

Figura 7.10 - Função "strtoval" [19]

Instructions ***MoveL***

MoveL **Moves the robot linearly**

MoveL is used to move the tool centre point (TCP) linearly to a given destination. When the TCP is to remain stationary, this instruction can also be used to reorientate the tool.

Example

MoveL p1, v1000, z30, tool2;

The TCP of the tool, *tool2*, is moved linearly to the position *p1*, with speed data *v1000* and zone data *z30*.

MoveL *, v1000\T:=5, fine, grip3;

The TCP of the tool, *grip3*, is moved linearly to a fine point stored in the instruction (marked with an *). The complete movement takes *5* seconds.

Arguments

MoveL [\Conc] ToPoint Speed [\V] | [\T] Zone [\Z] Tool [\WObj] [\Corr]

Figura 7.11 - Função "movel" [19]

Functions ***Offs***

Offs **Displaces a robot position**

Offs is used to add an offset to a robot position.

Examples

MoveL Offs(p2, 0, 0, 10), v1000, z50, tool1;

The robot is moved to a point *10* mm from the position *p2* (in the z-direction).

p1 := Offs (p1, 5, 10, 15);

The robot position *p1* is displaced *5* mm in the x-direction, *10* mm in the y-direction and *15* mm in the z-direction.

Return value Data type: *robtarget*

The displaced position data.

Arguments

Offs (Point XOffset YOffset ZOffset)

Figura 7.12 - Função "offs" [19]

Anexo III – Documentação do Robot IRB 140 e Controlador IRC 5

Specification			
Robot versions	**Handling capacity**	**Reach of 5th axis**	**Remarks**
IRB 140/IRB 140T	6 kg	810 mm	
IRB 140F/IRB 140TF	6 kg	810 mm	Foundry Plus 2 Protection
IRB 140CR/IRB 140TCR	6 kg	810 mm	Clean Room
IRB 140W/IRB 140TW	6 kg	810 mm	Wash Protection
Supplementary load (on upper arm alt. wrist)			
on upper arm		1 kg	
on wrist		0.5 kg	
Number of axes			
Robot manipulator		6	
External devices		6	
Integrated signal supply	12 signals on upper arm		
Integrated air supply	Max. 8 bar on upper arm		
IRC5 Controller variants:	Single cabinet, Dual cabinet, Compact, Panel mounted		

Figura 7.13 - Especificações do robot IRB 140 [20]

Performance		
Position repeatability	0.03 mm (average result from ISO test)	
Axis movement	**Axis**	**Working range**
	1, C Rotation	360°
	2, B Arm	200°
	3, A Arm	280°
	4, D Wrist	Unlimited (400° default)
	5, E Bend	240°
	6, P Turn	Unlimited (800° default)
Max. TCP velocity		2.5 m/s
Max. TCP acceleration		20 m/s²
Acceleration time 0-1 m/s		0.15 sec

Figura 7.14 - Alcance máximo de cada eixo do robot [20]

Specification	
Control hardware:	Multi-processor system PCI bus Flash disk for mass memory Energy back-up power failure handling USB memory interface
Control software:	Well proven real-time OS High-level RAPID programming language PC-DOS file format Preloaded software, also available on DVD Extensive functionality set, see separate RobotWare data sheet
Electrical Connections	
Supply voltage:	200-600 V, 50-60 Hz Integrated transformer or direct mains connection

Figura 7.15 - Especificações do controlador IRC5 [21]

Physical	Size H x W x D	Weight
Single cabinet	970 x 725 x 710 mm	150 kg
Dual cabinet	1370 x 725 x 710 mm	180 kg
Control module	720 x 725 x 710 mm	50 kg
Drive module	720 x 725 x 710 mm	130 kg
Empty cabinet for customer equipment	- small 720 x 725 x 710 mm - large 970 x 725 x 710 mm	35 kg 42 kg
Panel Mounted		
Control module	375 x 498 x 271 mm	12 kg
Drive Module	375 x 498 x 299 mm	24 kg
Compact controller	258 x 450 x 565 mm	27.5 kg

Figura 7.16 - Dimensões do controlador [21]

Anexo IV- especificações do robot IRB140, controlador IRC5 e Flexpendant

Robot IRB 140

O IRB 140 da ABB, figura 7.17, é um robot industrial utilizado em inúmeras tarefas. Este possui 6 eixos rotacionais que lhe dão grande mobilidade. Com uma carga máxima de 6 kg e com alcance máximo de 810 mm tornam um dos robots mais requisitados pela indústria [20].

Figura 7.17 - Robot IRB 140 da ABB

Sendo um dos mais pequenos da sua gama e utilizado em várias tarefas, podem ser os motivos para a sua escolha. Este robot é, na maioria das vezes, utilizado como soldador, limpeza, pintura, armazenamento e rebarbador [20].

Na figura 7.18 é possível verificar onde se encontram os eixos do robot IRB 140.

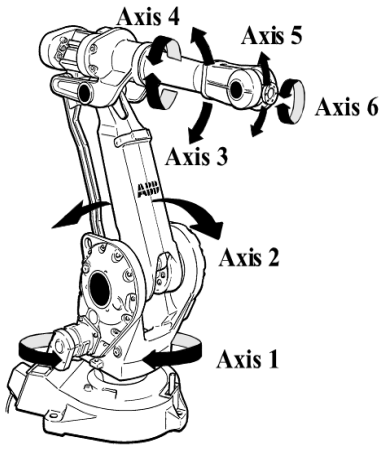


Figura 7.18 - Eixos do robot

Cada um dos eixos da figura 7.18 são capazes de exercer movimento rotacional, estando limitados a ângulos máximos de forma a serem evitadas colisões. Na tabela 7.1 pode-se verificar esses ângulos que as suas juntas são capazes de fazer.

Tabela 7.1 - Ângulos máximos de cada eixo [20]

Eixo	***Ângulo (°)***
1	360°
2	200°
3	280°
4	Ilimitado (400° por defeito)
5	240°
6	Ilimitado (800° por defeito)

Este robot é extremamente rápido. Podendo, um dos seus eixos, atingir os 450°/s, faz com que este robot seja ideal para realizar processos industriais. Através da tabela 7.2 é possível verificar as velocidades de cada eixo.

Tabela 7.2 - Velocidades dos eixos do robot [20]

Eixo	*Velocidade (°/s)*
1	200°/s
2	200°/s
3	260°/s
4	360°/s
5	360°/s
6	450°/s

As suas dimensões podem ser consultadas na tabela 7.3.

Tabela 7.3 - Dimensões do robot [20]

Braço	*Dimensão (mm)*
1	352 mm
2	360 mm
3	240 mm
4	140 mm
5	65 mm

Controlador IRC 5

O IRC 5, figura 7.19, é um controlador dos robots da ABB que trás flexibilidade e segurança à utilização deste tipo de tecnologia. Sempre que se queira utilizar um robot é imprescindível o uso de um controlador, pois serão aí recebidos, interpretados e enviados sinais pelo utilizador/programador para o controlo do robot.

Figura 7.19 - Controlador IRC 5

O controlador IRC 5 é extremamente seguro, pois existem alguns dispositivos que permitem o bloqueio dos motores do robot para este não magoar ou danificar pessoas e equipamentos.

O controlador permite a programação do robot em linguagem RAPID, esta linguagem permite a utilização de comandos especiais associados ao robot, simplificando a sua programação [21].

Em termos de comunicações o IRC 5 permite comunicação via porta série, explicada anteriormente, e FTP. A ligação FTP serve para estabelecer contacto entre o IRC 5 e o software *RobotStudio On-line* [21].

Seguramente, a sua ferramenta mais importante é a possibilidade de utilização da Flexpendant que serve para a interacção entre o utilizador e o robot.

No lado esquerdo do controlador IRC 5 existem uma série de botões designados de Botões de módulo de controlo, Figura 7.20 [22].

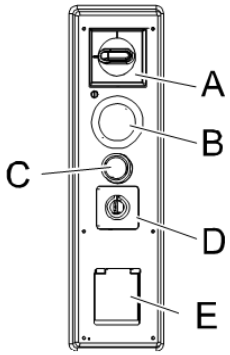


Figura 7.20 - Botões de módulo de controlo

Tabela 7.4 - Botões do módulo de controlo [22]

Botão	***Legenda***
A	Alimentação ON/OFF
B	Botão de emergência
C	Motores ligados
D	Chave de modo
E	USB e FTP

O botão de alimentação (A) serve para ligar/desligar o sistema, o botão de emergência (B) pára o robot quando pressionado, o botão motor ligado (C) serve para ligar/desligar os motores do robot, em (E) são as portas para ligação USB e FTP.

O botão de chaves de modo (E), figura 7.21, serve para escolher qual o tipo e velocidade de funcionamento do robot.

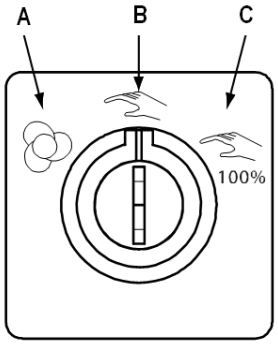


Figura 7.21 - Chave de modo

Esta opção (figura 7.21) é apenas acedida através de uma chave do módulo de controlo.

Tabela 7.5 - Legenda da chave de modo

Botão	***Legenda***
A	Modo automático
B	Modo manual com velocidade reduzida
C	Modo manual com velocidade total

A porta série, assim como outros dispositivos, encontra-se dentro do armário de controlo do IRC 5 (figura 7.22).

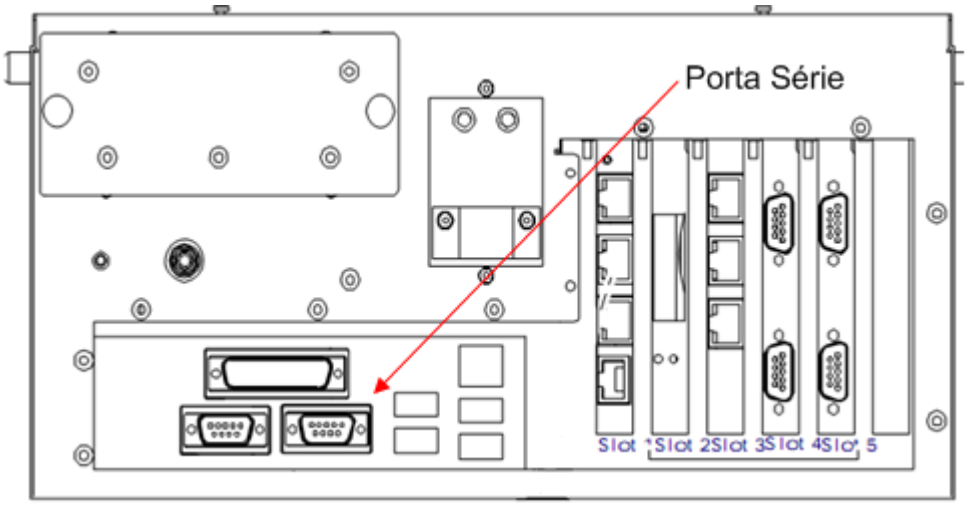


Figura 7.22 - Imagem do interior do armário

Por fim, o IRC 5 tem a dimensão de 1370x725x710 mm e pesa 180 kg, não sendo nem em peso ou em tamanho semelhante a algumas máquinas industriais utilizadas.

Flexpendant

O Flexpendant, figura 7.23, é um dispositivo de interacção entre o robot e o utilizador/programador. Com este mecanismo é possível programar e realizar operações envolvidas com o sistema robótico [22].

O Flexpendant é constituído por diversos acessórios, entre os mais importantes estão os botões, joystick, conector e ecrã táctil.

Figura 7.23 - Flexpendant

Os acessórios da consola Flexpendant podem ser vistos na figura 7.24.

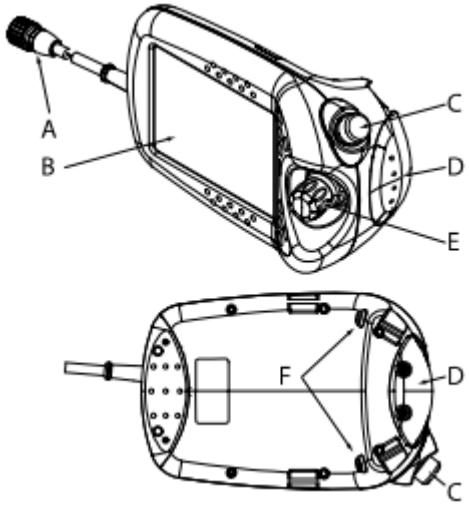


Figura 7.24 - Acessórios da Flexpendant [22]

A legenda da figura 7.24 encontra-se na tabela 7.6.

Tabela 7.6 - Legenda dos acessórios da Flexpendant [22]

Peça	***Legenda***
A	Conector
B	Ecrã táctil
C	Botão de emergência
D	Dispositivo de activação

E	Joystick
F	Botões *hold-to-run* (apenas disponíveis em alguns sistemas)

O dispositivo de activação (D) é um mecanismo de segurança, sempre que este é pressionado os travões do robot são libertados e o robot está pronto para iniciar o movimento. Se o botão for libertado os travões são activados e o robot cessa a actividade.

Existem, também, alguns botões na consola táctil Flexpendant, Figura 7.25.

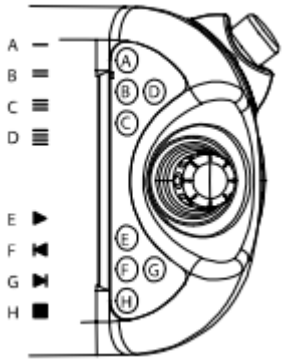


Figura 7.25 - Botões da Flexpendant [22]

A legenda da figura 7.25 encontra-se na tabela 7.7.

Tabela 7.7 - Legenda dos botões da Flexpendant [22]

Peça	***Legenda***
A	Botão programável 1
B	Botão programável 2
C	Botão programável 3
D	Botão programável 4
E	Executar programa
F	Retroceder no programa passo-a-passo

G	Avançar no programa passo-a-passo
H	Parar execução

Os botões programáveis de 1 a 4 são associados a acções do robot, como aceder as suas entradas e saídas digitais [22].

É no ecrã táctil, figura 7.26, onde o utilizador/programador opera.

Figura 7.26 - Ecrã inicial Flexpendant [22]

Na tabela 7.8 encontra-se a legenda da figura 7.26.

Tabela 7.8 – Legenda do ecrã inicial da Flexpendant

Botão	***Legenda***
A	Menu ABB
B	Barra de status
C	Página principal
D	Fechar
E	Barra de tarefas
F	Menu *Quickset*

Para criar um programa com a consola táctil Flexpendant, basta clicar no menu ABB (A) da figura 7.26, onde aparecerá um submenu, figura 7.27, com várias opções para o utilizador/programador escolher.

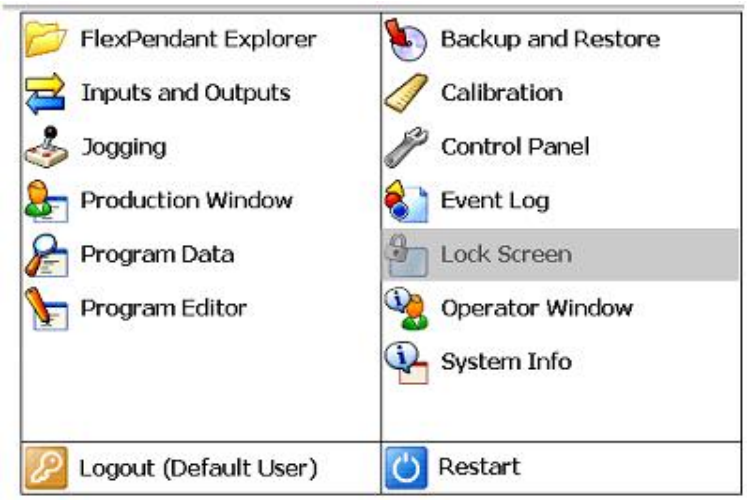


Figura 7.27 - Opções do menu ABB

Clicando na opção *Program editor*, é aberta nova tela, onde será introduzido código RAPID para controlar o robot, Figura 7.28.

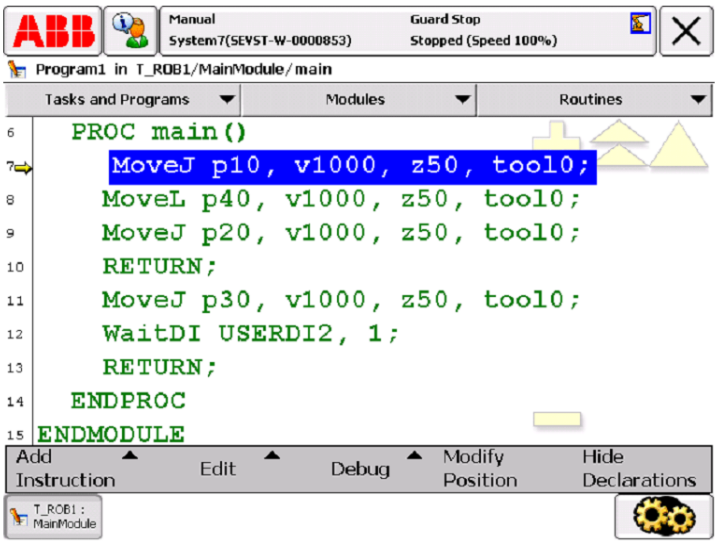


Figura 7.28 - Editor de programa

Como se pode verificar existe uma opção de adição de instruções (*Add Instruction*) onde todo o código é colocado com a ajuda dessa ferramenta. Todas as instruções podem ser alteradas através da opção *Edit*, a consola permite também do *Debug* do código introduzido. Com ajuda destas ferramentas é impossível existirem erros de sintaxe no código introduzido, ajudando o programador a não desperdiçar tempo ao longo da programação.